# VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF CONTROL AND INSTRUMENTATION


# VYUŽITÍ ROBOTICKÉHO MANIPULÁTORU PRO SIMULACI STISKŮ DOTYKOVÉHO DISPLEJE

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

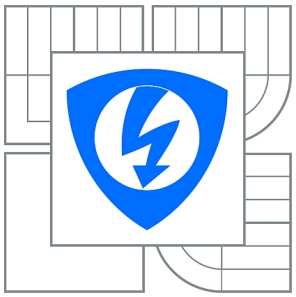

AUTOR PRÁCE ONDŘEJ TATÝREK
AUTHOR



BRNO 2014


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY


FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF CONTROL AND INSTRUMENTATION


# VYUŽITÍ ROBOTICKÉHO MANIPULÁTORU PRO SIMULACI STISKŮ DOTYKOVÉHO DISPLEJE

USING ROBOTIC MANIPULATOR FOR TOUCHSCREEN PRESSING SIMULATION

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS


AUTOR PRÁCE — ONDŘEJ TATÝREK

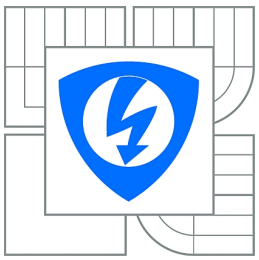
AUTHOR

VEDOUCÍ PRÁCE — Ing. ADAM CHROMÝ

SUPERVISOR



BRNO 2014



**VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ**

**Fakulta elektrotechniky a komunikačních technologií**

**Ústav automatizace a měřicí techniky**

# Bakalářská práce

bakalářský studijní obor
**Automatizační a měřicí technika**

| | | | |
|---|---|---|---|
| ***Student:*** | Ondřej Tatýrek | ***ID:*** | 146115 |
| ***Ročník:*** | 3 | ***Akademický rok:*** | 2013/2014 |

**NÁZEV TÉMATU:**

## Využití robotického manipulátoru pro simulaci stisků dotykového displeje

**POKYNY PRO VYPRACOVÁNÍ:**

Cílem práce je vytvoření funkčního zařízení umožňující testování dotykových displejů dle předem nadefinovaných testů. Zadání lze shrnout do následujících bodů:
1. Analyzujte principy a vlastnosti robotických manipulátorů a dotykových displejů.
2. Navrhněte a realizujte dotykový efektor schopný simulovat dotyk prstu na rezistivním i kapacitním displeji.
3. Vytvořte třídu, která naplánuje trajektorii manipulátoru tak, aby se dotykový efektor dotkl všech zadaných bodů.
4. Rozšiřte předchozí třídu o možnost zadávání souřadnic ve formě pozice v 2D obraze víme-li, pod jakým směrovým vektorem je obraz snímán.
5. Vytvořte přehledné uživatelské rozhraní umožňující definovat a spouštět jednotlivé sekvence dotyků.
Navržené řešení bude použito při vývoji automatizovaných testů uživatelského rozhraní multifunkčních tiskáren ve společnosti Y Soft Corporation, a.s.

**DOPORUČENÁ LITERATURA:**

A. Palko, Technické prostriedky pre automatizáciu výrobných procesov - Robotika, 2010, ISBN 9788071658078

| | | | |
|---|---|---|---|
| ***Termín zadání:*** | 10.2.2014 | ***Termín odevzdání:*** | 26.5.2014 |

***Vedoucí práce:*** Ing. Adam Chromý
***Konzultanti bakalářské práce:***

**doc. Ing. Václav Jirsík, CSc.**
*Předseda oborové rady*

## ABSTRAKT

Bakalářská práce se zabývá volbou a popisem robotického manipulátoru pro simulaci doteků. Je rozebrán návrh tvaru, volba materiálu a výroba efektoru pro simulaci stisku dotykového displeje prstem. V práci je popsána uživatelská aplikace, která umožňuje definici, úpravu a spouštění testů dotykových displejů společně s vyhodnocením obrazu. Jsou popsány významné metody programu, jejich fungování a provázanost s ostatními metodami.

## KLÍČOVÁ SLOVA

Simulace stisků dotykového displeje prstem, manipulátor Epson C3, efektor pro dotek displeje, automatizované testování,

## ABSTRACT

This bachelor thesis is about choice and description of robotic manipulator for simulation finger touch on touchscreen. Describes blueprint of shape, choice of material and made of effector. Also describes user aplication, which enables to define, edit and run test of touchscreen together with image evaluation. Most importent methods are described and explained how they works inside.

## KEYWORDS

Simulation of finger touch on touch screen, manipulator Epson C3, touchscreen effector, automated testing

TATÝREK, Ondřej *Využití robotického manipulátoru pro simulaci stisků dotekového displeje*: bakalářská práce. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřící techniky, 2014. 59 s. Vedoucí práce byl Ing. Adam Chromý,

# PROHLÁŠENÍ

Prohlašuji, že svou bakalářskou práci na téma „ Využití robotického manipulátoru pro simulaci stisků dotekového displeje“ jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ..... .....

(podpis autora)

# PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu semestrální práce panu Ing. Adamovi Chromému za odborné vedení, konzultace, trpělivost, cenné rady a podnětné návrhy k práci.


Brno ..... .....

(podpis autora)

# OBSAH

# SEZNAM OBRÁZKŮ

# ÚVOD

Automatizace testování výrobků se dnes stává velmi oblíbené a výrobci čím dál více praktikované. Výhody plynoucí ze strojového testování výrobků jsou zřejmé. Všechny výrobky budou vystaveny naprosto identickému testu, a proto si můžeme být více jistí kvalitou testování. Výhodou je rychlost testování. Zpravidla strojový test bude proveden rychleji, než kdyby kontrolu prováděl člověk. I když mohou být výrobní procesy, kde je člověk schopen vyhodnocení provést mnohem rychleji, u strojů nehrozí chyba například v důsledku únavy nebo nesoustředěnosti. Nespornou výhodou je i úspora nákladů při nasazení takového postupu při masové výrobě.

V našem konkrétním případě je vytvořen automatizovaný tester komunikačního rozhraní dotykových displejů u multifunkčních tiskáren ovládaných pomocí dotykového displeje. Výhodou těchto tiskařských strojů je opravdová multifunkčnost a pokročilé možnosti nastavení tisku jako je oboustranný tisk, tisk knih, řazení listů dle požadavků, scanování, kopírování a podobně. Právě kvůli vysokému množství funkcí, které takové zařízení sdružuje, je zapotřebí dostatečně otestovat, zda softwarová tlačítka na displeji pracují správně a na displeji i v zařízení se děje to, co skutečně chceme. I když v této práci je rozebrán návrh zařízení pro testování dotykových displejů multifunkčních tiskáren, je možné jej použít pro testování dotykových displejů obecně.

Testování je možné provádět několika způsoby, například připojení zařízení k PC, kde pomocí simulátoru budeme do tiskárny zasílat požadavky. Požadavkem této práce je ale testovat komunikační rozhraní člověk-stroj. Je použit dotykový efektor jako náhrada lidského prstu a vyhodnocení je prováděno za pomoci kamery a zpracování jejích snímků. Výsledkem tedy bude míra shody obrazu z kamery a vzorového obrazu ze stejné kamery, ale pořízeném při definici testu. Zadáním celé práce je tedy vytvoření efektoru pro ovládání displeje a ovládací aplikace, která umožňuje zadání vytvoření a uložení nového testu, úpravu stávajících testů, jejich náhled, spuštění a vyhodnocení. V této práci jsou popsány všechny zmíněné části, mimo vyhodnocení, respektive porovnávání shody dvou obrazů. Tato část je zadáním samostatné bakalářské práce, ale v závěru budou výsledky obou bakalářských prací sloučeny v jeden výsledný produkt. Jako pohyblivý aktor je zvolen průmyslový manipulátor EPSON C3. V první části je problematika manipulátorů, dotykových displejů, homogenní transformace a popis zvolené kamery. Ve druhé části této práce je rozebrána konstrukce koncového členu pro manipulátor, který se bude displeje dotýkat. Třetí se věnuje ovládací aplikaci a jejímu popisu.

# 1 TEORETICKÝ ÚVOD

## 1.1 Robotický manipulátor

### 1.1.1 Požadavky

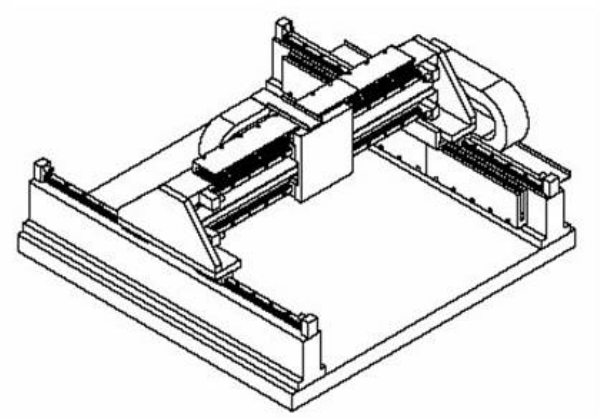
V naší aplikaci je k robotickému manipulátoru připevněn efektor společně s kamerou, která snímá obraz před efektorem. Prvním požadavkem tedy je, aby na manipulátor bylo možné uchytit vlastní koncový člen. S tímto členem musí být schopen pohybovat s dostatečnou přesností, a tedy musí být dostatečně přesný v řízení polohy koncového bodu. Pro tuto aplikaci je přesnost +- 0,5 mm vyhovující, a proto je třeba najít manipulátor, který disponuje alespoň touto přesností. Také rozsah pohybu musí být dostatečný, aby byl schopen obsloužit zařízení, které nemusí být vždy těsně u manipulátoru, ale muže se nacházet v prostoru několik desítek centimetrů vzdáleném a také ovládaný panel může mít různou velikost, k čemuž je třeba přihlédnout. Jelikož vždy nebude rovina doteku shodná s pracovní rovinou manipulátoru, ale bude různě nakloněna v prostoru, bude nutné, aby manipulátor byl schopen koncový člen naklánět do stran a zajistit tak, aby se efektor dotýkal displeje kolmo. Rychlost manipulátoru není pro tuto aplikaci příliš podstatná. Vzhledem k povaze aplikace, pro kterou je manipulátor využit, je vhodný kandidát vyhledáván pouze mezi stacionárními manipulátory, jelikož pohyblivost není vyžadována.

### 1.1.2 Rozdělení manipulátorů

#### Portálový manipulátor

Portálový manipulátor standardně disponuje třemi stupni volnosti a pohybuje se v kartézském souřadném systému ve směru os [x,y,z]. Portálový manipulátor je na obrázku 1.1. Konstrukce se skládá ze dvou horizontálních a jednoho vertikálního posuvu. Horizontální jsou pevné a vertikální muže být buď také pevný, nebo je nahrazen lanem. Velkým omezením je to, že se třemi stupni volnosti není manipulátor schopen polohovat koncový bod žádným úhlovým natočením, což je jedním z požadavků kladených na robota. Navíc jelikož nejsou schopny dosáhnout krajních poloh, je jejich manipulační prostor vždy menší než jsou rozměry celého zařízení [7]. Mohl by být ale použit takový portálový manipulátor, který by odstranil některé z nevýhod. Také by mohl být vytvořen takový portálový manipulátor, který by měl malé rozměry, lehkou a přenosnou konstrukci a tím i nízkou cenu. Takovýto manipulátor by byl posazen přímo nad displej, který bude ovládat. Tím bude jednak zjednodušeno definování výchozí pozice robotu a zároveň bude zajištěno, že manipulační

rovina displeje a manipulátoru jsou shodné, což zjednoduší program pro ovládání. [15]



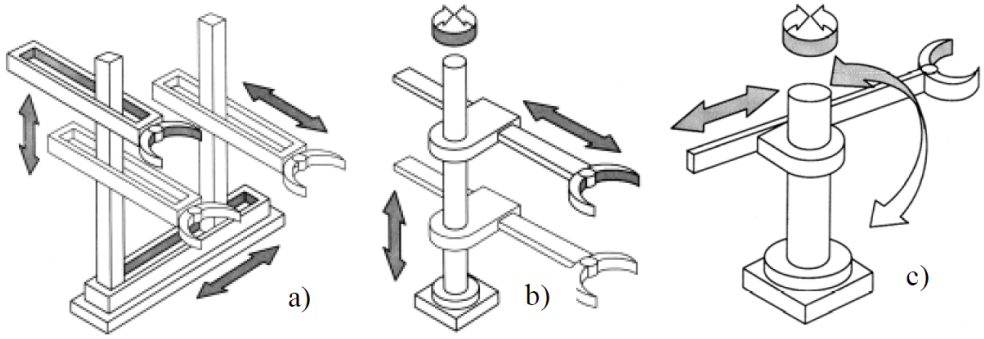
Obr. 1.1: Portálový manipulátor [3]

### Manipulátor s ramenem

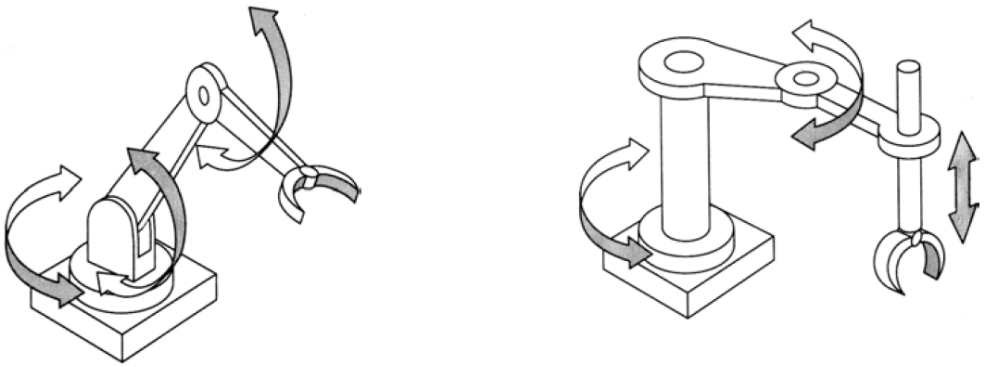
Jinou možnou konstrukcí robotického manipulátoru je konstrukce podobná lidské paži. Je tedy podobně jako spousta jiných robotických zařízení a konstrukcí inspirovaná studiem přírody. Konstrukcí tohoto typu existuje celá řada. Některé z nich jsou na obrázcích 1.2 a 1.3. Všechny tyto koncepce disponují třemi stupni volnosti (tři polohově nastavitelné klouby), a přitom každá má určité výhody a nevýhody.

Manipulátor a) na obrázku 1.2 nemá žádný otočný kloub ale má posuvy ve všech třech směrech. Takovýto robot by byl vhodný například pro ukládání předmětů do regálů ve skladišti. Kvůli tomu, že se nemůže otáčet musí být vždy ukládaná věc dopravena až před robota. V případě b) a c) jsou manipulátory schopny otáčení, ale nemají možnost pohybu do strany. Manipulátory z obrázku 1.3 mají lepší manipulační schopnost v některých osách, v jiných ale horší.

Z toho tedy vyplývá, že nezáleží jen na počtu stupňů volnosti, ale i na jejich vhodném uspořádání. Ve specifických aplikacích nemusí být zapotřebí ani posuv ve třech osách jako je například posuvník, který využívá pohyb jen v jedné ose. Každý z uvedených konceptů najde využití ve specifických aplikacích, ale pro potřeby této práce každý přináší určitá omezení a proto není vhodný ani jeden, jelikož je zapotřebí vyšší flexibilita stroje. Je nutné zvolit manipulátor s vyšším počtem stupňů volnosti. Zároveň je ale nutné si uvědomit, že čím vyšším počtem stupňů volnosti manipulátor disponuje, tím vyšší je nepřesnost manipulace koncového bodu. Pokud tato má být zachována, je nutno použít přesnější efektory pohybu a se zvyšující se přesností velmi rychle narůstá i cena. [4]




Obr. 1.2: Manipulátor s ramenem [4]

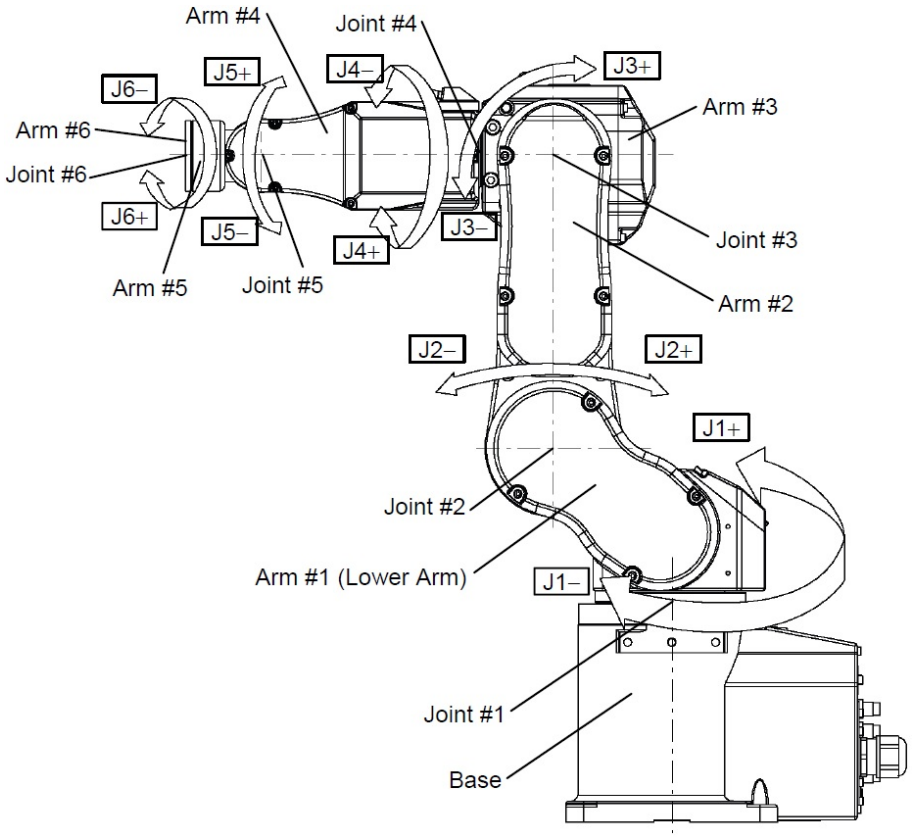


Obr. 1.3: Manipulátor s ramenem [4]

### 1.1.3 Zvolený manipulátor

Vybraným manipulátorem se stal robot značky Epson s označením C3. Tento stroj je zvolen především z důvodu, že jej škola již vlastní a není tedy zapotřebí další investice. Velkou předností manipulátoru je především vysoká variabilita ramene tak, jak je naznačeno na obrázku 1.4. Vysokou manipulovatelností disponuje díky šesti stupňům volnosti ramene. Pokud tedy zadáváme souřadnice bodu, do kterého se má rameno dostat, nestačí pouze hodnoty souřadnic [x,y,z], ale musí být zadána i úhlová orientace koncového bodu pomocí úhlových souřadnic [$\alpha$, $\beta$, $\gamma$] [4]. Robot dokáže manipulovat koncový bod s přesností +- 0,02 mm [5], což je mnohem vyšší přesnost, než by pro dané využití postačovala. Velkou výhodou je také velikost celého zařízení a s ní spojená velikost manipulačního prostoru. Do tohoto prostoru by bylo možné připravit i více zařízení a vhodnou algoritmizací obsluhovat i více zařízení najednou. Ovšem pokud by měl být manipulátor znovu pořizován, byl by tento robot značky Epson nevhodný především z důvodu velmi vysoké ceny zařízení. Dle mého názoru by byla nejvhodnější volba malého, lehkého, přenosného a levného

portálového manipulátoru, který by bylo možné přichytit přímo k testovanému zařízení. Další výhodou by bylo to, že takovýto malý manipulátor by pravděpodobně nebyl schopen vyvinout takovou sílu, aby zařízení poškodil nebo zničil. To ale zvolenému robotu prohlásit nelze, a proto musí být kladen velký důraz na bezpečnost při obsluze.




Obr. 1.4: Rameno robotu Epson [5]

## 1.2 Principy funkce dotykových displejů

Dotykové displeje jsou v poslední době čím dál častěji využívané jako vstup pro komunikaci s různými zařízeními. Jejich největší výhodou je jejich obrovská variabilita a to nejen ve velikosti displeje, ale především možnost zobrazení ovládacích prvků přímo na displeji. Tím jsou nahrazována jednoúčelová hardwarová tlačítka a je tak ušetřeno místo na desce plošných spojů. Také se na displeji zobrazí pouze ty obslužné prvky, které jsou právě zapotřebí a tím je i obsluha daného zařízení přehlednější a jednodušší.

Pro snímání místa dotyku na displeji bylo vyvinuto několik různých technologií pracujících na různých principech. Dnes nejhojněji využívané jsou rezistivní a kapacitní technologie, ale existují i další, jako jsou dotykové displeje s infračerveným zářením, nebo displej s povrchovou akustickou vlnou. [9]

### 1.2.1 Rezistivní technologie

Rezistivní technologie je z pohledu jejího objevu starší technologií. Řez takovýmto displejem je na obrázku 1.2.1. Je zde vidět základní nosná vrstva, nejčastěji skleněná, na ní tenká vrstva vodičů, zeleně nakreslené distanční body a svrchní pružnou vrstvu, která je ze své spodní strany vodivá.

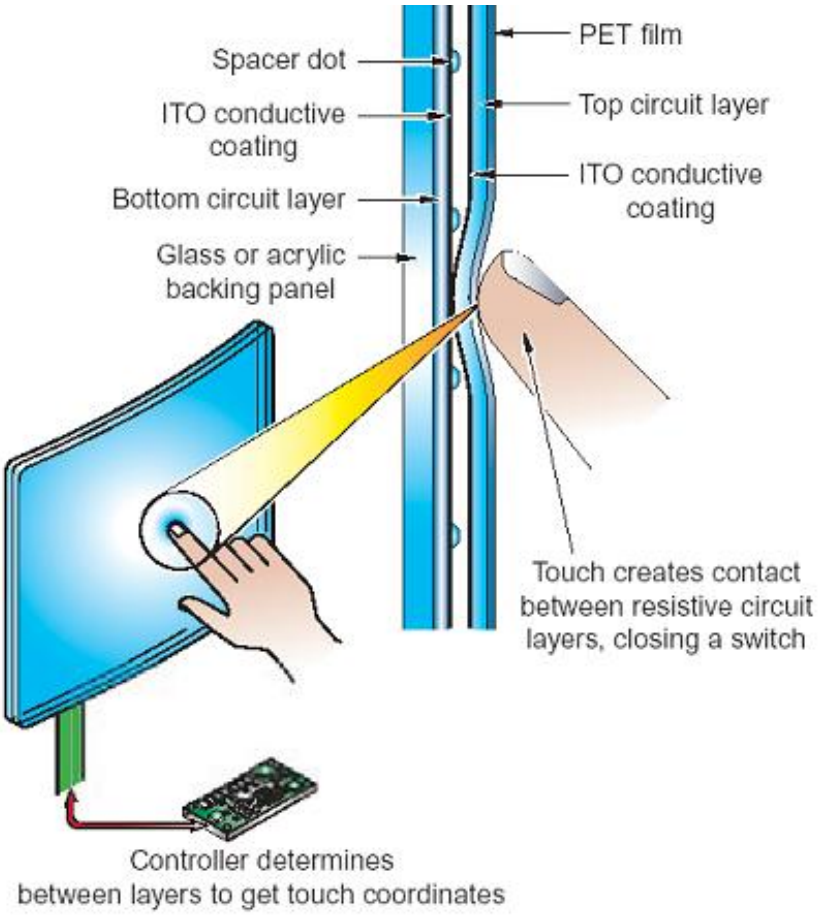
Princip funkce displeje je takový, že v okamžiku doteku (při působení tlakem na displej) dojde ke spojení spodní a horní vodivé vrstvy. V důsledku tohoto spojení začne obvodem protékat proud. Místo doteku je vyhodnoceno kontrolerem, který je k vodičům displeje připojen. Ten vyhodnotí, na kterém vodiči došlo ke spojení a změřením impedance smyčky určí i vzdálenost spojení na daném vodiči. Tím je přesně určeno místo doteku. To, aby vodiče nebyly spojeny při malém přítlaku a nedocházelo tak k přeslechům, je zabezpečeno pomocí distančních dielektrických podpěr, které drží svrchní pružnou vrstvu v dostatečné vzdálenosti od vodičů dolní vrstvy.

Díky tomu, že dotykové displeje s resistivní technologií reagují na tlak, je možné je ovládat téměř jakýmkoli předmětem. Samozřejmě nesmí být předmět natolik ostrý, aby způsobil poškození měkké horní vrstvy. Jednou z nevýhod této technologie je skutečnost, že vrstvy napařeného kovu snižují propustnost světla na úroveň kolem osmdesáti procent.

Technologie má navíc dvě různá provedení. Čtyřvodičové, kde je napájení přivedeno do spodní vrstvy, a pětivodičové, ve kterém je napájena horní pružná vrstva. Ve druhém případě je tak dosaženo vyšší odolnosti a udávaná odolnost je třicet pět milionů doteků v jednom místě. [9, 10]

### 1.2.2 Kapacitní technologie

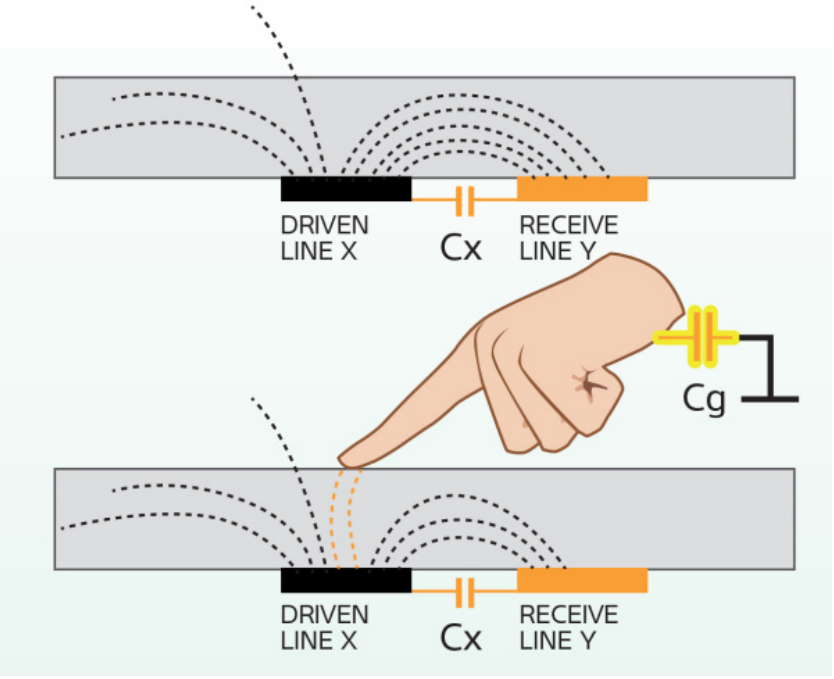
Jedná se o dražší technologii než je rezistivní. U kapacitních dotykových displejů je využito přirozené vodivosti lidského těla, popřípadě jiných vodivých materiálů, které způsobí dostatečnou změnu kapacity pro vyhodnocení dotyku. Po displeji jsou taženy vodiče ve dvou směrech vzájemně na sebe kolmých. Mezi těmito vodiči existuje kapacitní vazba, která je dotekem vodivého předmětu narušena a část přenášeného náboje je svedena pryč (obr. 1.2.2). Vyhodnocení spočívá ve zjištění, mezi kterými




Obr. 1.5: Princip funkce rezistivního displeje [9]

dvěma vodiči došlo ke změně kapacity. Díky známé poloze vodičů je snadné určit jejich průsečík a tím i bod dotyku.

Hlavní nevýhodou oproti rezistivní technologii je nutnost vodivosti předmětu, který se dotýká displeje. Existují však speciální stylusy pro kapacitní displeje. Naopak výhodou oproti rezistivním displejům je vyšší propustnost světla, pohybující se okolo devadesáti tří procent. Další výhodnou vlastností je vyšší mechanická odolnost díky svrchní vrstvě vyrobené ze skla nebo jiného pevného nevodivého materiálu. Navíc disponují vyšším rozlišením místa dotyku a i přesto je u této technologie udávána odolnost až tři sta milionů doteků v jednou místě. V průmyslových aplikacích jsou více využívány rezistivní displeje. [9, 10]




Obr. 1.6: Princip funkce kapacitního displeje [10]

## 1.3 Kamera

### 1.3.1 Požadavky na kameru

Hlavním požadavkem na kameru je zprostředkování obrazu. Není ale zapotřebí, aby se jednalo o vysokorychlostní kameru, jelikož její obraz bude využíván pouze pro ukládání snímku v určitých časových okamžicích, které jsou od sebe velmi vzdáleny (jednotky sekund). Pro vyhodnocení obrazu bude vhodnější kamera s barevným obrazem, který musí mít dostatečné rozlišení. To především z důvodů, že je velmi pravděpodobné, že v obrazu budou hrát významnou roli malé detaily jako jsou popisky a obrázky ikon, zatím co zbytek scény bude identický. Důraz musí být kladen také na optiku kamery. Ta musí být schopna vyostřit obraz již od několika centimetrů, aby byl displej pokud možno zobrazen v co největší ploše snímku.

### 1.3.2 Zvolená kamera

Je zvolena kamera od výrobce The Imaging Source, nesoucí označení DKF 51BG02.H. Jedná se o makrokameru s vysokým rozlišením a barevným obrazem. Její parametry jsou uvedeny v následujícím seznamu.

Parametry kamery IS DKF 51BG02.H

- Nativní rozlišení - 1600x1200

- Počet snímků za sekundu - až 15
- Výstupní formát dat - Y800, RGB32
- Světelnost optiky - 1,2
- Ostření - 19,6 cm - nekonečno
- Čočka objektivu - 6 mm

Daná kamera je zvolená především z důvodu, že ji škola již vlastní, ale jak je možné dle jejích parametrů posoudit, je skutečně vhodná pro tuto aplikaci. [12]


Obr. 1.7: Kamera Imaging Source DFK 51BG02.H [11]

### 1.3.3 Popis kalibrace kamery

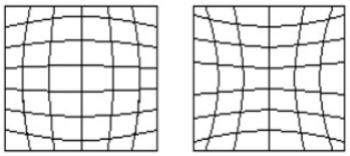
Jelikož zvolená kamera má objektiv se širokým záběrem (objektiv typu 'rybí oko'), dochází ke zkreslení obrazu. Toto zkreslení působí problémy při definici nového testu, jelikož operátor vybírá body kliknutím do obrazu, který neodpovídá skutečnému stavu. Při klikání do obrazu jsou souřadnice myši snímány v ortogonální mřížce. Z deformovaného obrazu kamery ale nelze přímo vyčítat hodnoty ortogonálních souřadnic. Z tohoto důvodu je třeba určit transformační vztah, pomocí kterého bude provedeno mapování souřadnic ze souřadnic obrazu v okně programu na reálné, ve kterých se manipulátor pohybuje.

Kalibrace kamer na zkreslení tohoto druhu (typu soudek / poduška) se provádí běžně a existuje obecný vzorec pro toto zkreslení (obr. 1.8). Pro výpočet skutečných souřadnic bodů v obrazu je zapotřebí vypočtení konstant, které dané zkreslení definují. Tato rovnice může obecně obsahovat více nebo i méně členů násobených konstantami zkreslení. To záleží podle toho jakou přesnost jeho definice je třeba

zajistit. Více členů znamená přesnější aproximaci, ale při použití právě třech členů je přesnost dostačující a výpočetní náročnost není přehnaně vysoká (lze vypočítat i ručně v rozumném čase).

Pro kalibraci zkreslení této kamery byla použita rovnice se třemi členy definujícími zkreslení. Pro určení těchto konstant zkreslení byl vytvořen kalibrační obrazec (obr. 1.9), který byl snímán kamerou a efektor byl nahrazen perem, které v kalibrační mříži označovalo body doteku. Jako body, ve kterých se měl robot dotknout byly vybírány všechny průsečíky černých čar. Po dokončení vykreslení bodů do mřížky bylo možné určit jednak posun (offset) celého obrazu, jako posun středů, a také z trojic bodů určit hodnoty konstant zkreslení dosazením ideálních a reálných (změřených) bodu do rovnic a řešit jejich soustavu. Řešení pomocí symbolického toolboxu matlabu ale nevedlo k cíli, jelikož pro většinu trojic zvolených rovnic neexistovalo řešení. To bylo způsobeno pravděpodobně drobnými odchylkami při vytváření kalibrační mříže a také chybou měření vzdálenosti bodů. Proto byla vytvořena v matlabu kriteriální funkce, které se snažila najít takové konstanty, aby odchylka změřených a vypočtených bodů byla minimální. Tato funkce byla použita opakovaně pro různé trojice bodů, ale výsledky nebyly jednoznačné. I když byly řádově stejné, i malý rozdíl konstant způsobil velkou změnu v deformaci obrazu. Proto byla nejvhodnější trojice konstant zvolena testováním v programu.

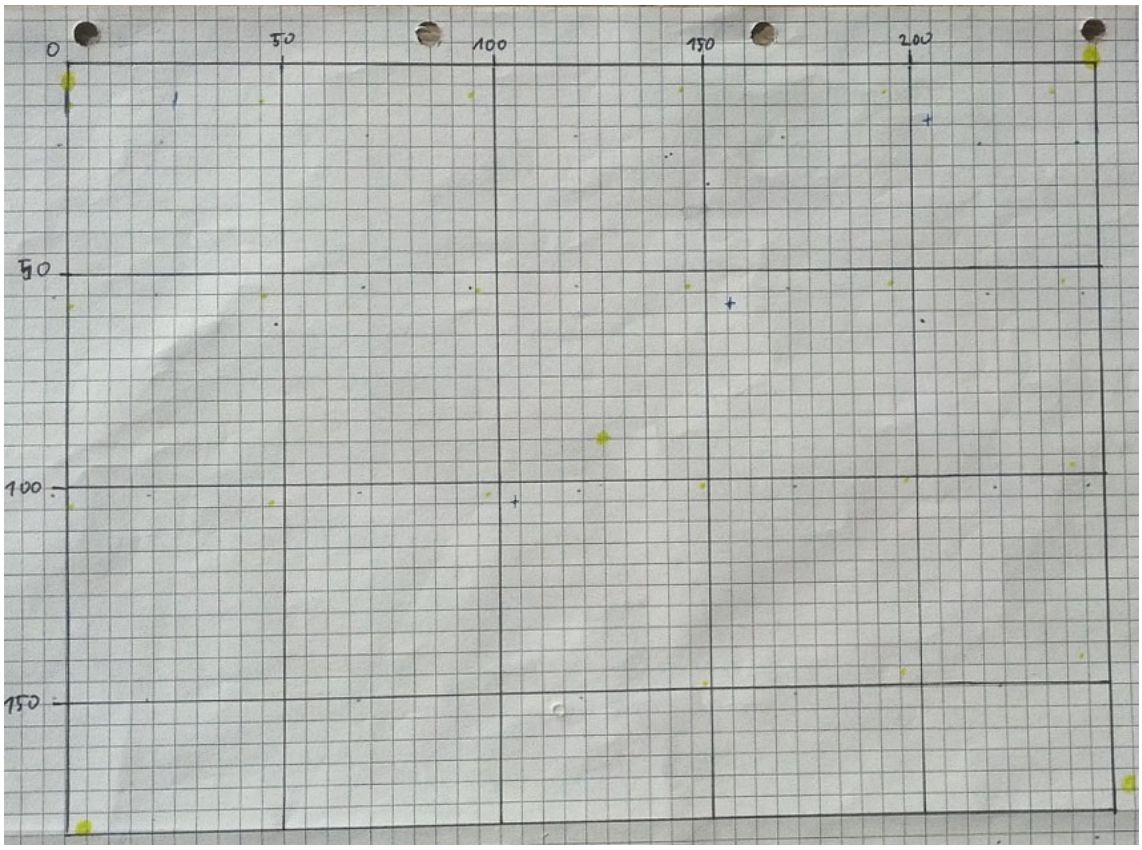
$$x' = x \cdot (1 + \kappa_1 \cdot r^2 + \kappa_2 \cdot r^4 + \kappa_3 \cdot r^6)$$
$$y' = y \cdot (1 + \kappa_1 \cdot r^2 + \kappa_2 \cdot r^4 + \kappa_3 \cdot r^6)$$



Obr. 1.8: Obecná rovnice deformace obrazu typu soudek/poduška [13]

## 1.4 Homogenní transformace

Pokud se práce nemá omezovat pouze na pohyby v základní rovině robotu, je nutné použít nějaký výpočetní vztah pro přepočet bodů ze základní roviny do roviny, ve které se nachází displej. Nová rovina je definována třemi úhly [U, V, W], které definují vychýlení os od základní roviny manipulátoru. Systém souřadnic může být kromě náklonu také posunut a toto posunutí je vypočteno od výchozího (vztažného) bodu, například souřadnice [0, 0, 0]. Homogenní transformace je taková transformace, která má jako vstup hodnoty souřadnic v základní rovině, posunutí a náklon nové roviny ve všech osách. Jejím výstupem jsou potom souřadnice, které určují bod nové roviny. Pokud je tento blok transformace přidán na vstup řízení manipulátoru,




Obr. 1.9: Kalibrační obrazec kamery

na jeho vstup jsou potom posílány souřadnice definované v základní rovině a robot pracuje v rovině nové (transformované).

Obecný tvar matice homogenní transformace je uveden na obrázcích 1.10 a 1.11. Celá matice je tedy složena ze čtyř matic. První je matice rotace v levém horním rohu o velikosti 3x3. Ta určuje natočení souřadného systému. V pravém horním rohu se nachází vektor translace (posunutí) o rozměrech 3x1. Ve spodním řádku je zahrnuta perspektiva a měřítko, což je využíváno v počítačové grafice. Pro naše využití však budou vždy stejné jako v základním tvaru rovnice. Přesný tvar rovnice implementovaný v programu je uveden na obrázku 1.12 a je převzat z práce pana Ing. Adama Chromého [14], který ve své diplomové práci řešil stejnou problematiku. [4]

$$\mathbf{H} = \begin{bmatrix} \textit{Matice rotace} & \textit{Vektor translace} \\ \textit{Perspektiva} & \textit{Měřítko} \end{bmatrix}$$

Obr. 1.10: Obecná rovnice homogenní transformace [4]

$$\mathbf{Trans}(a,b,c) = \begin{bmatrix} 1 & 0 & 0 & a \\ 0 & 1 & 0 & b \\ 0 & 0 & 1 & c \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Obr. 1.11: Základní tvar homogenní transformace (netransformuje) [4]

$$\begin{bmatrix} c(u_t)\ c(v_t) & -c(v_t)\ s(u_t) & s(v_t) & x_t \\ c(w_t)\ s(u_t) + c(u_t)\ s(v_t)\ s(w_t) & c(u_t)\ c(w_t) - s(u_t)\ s(v_t)\ s(w_t) & -c(v_t)\ s(w_t) & y_t \\ s(u_t)\ s(w_t) - c(u_t)\ c(w_t)\ s(v_t) & c(u_t)\ s(w_t) + c(w_t)\ s(u_t)\ s(v_t) & c(v_t)\ c(w_t) & z_t \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Obr. 1.12: Tvar homogenní transformace implementované v programu [14]

# 2 KONSTRUKCE KONCOVÉHO EFEKTORU

## 2.1 Koncový efektor

Tato části práce se zabývá konstrukcí dotykového efektoru pro robotický manipulátor. Nejprve je důležité ujasnit, jaké funkce má efektor plnit. Z požadavků na funkčnost je odvozen výsledný tvar efektoru, ale také je myšleno na co nejjednodušší konstrukci, aby ji bylo možné vyrobit s co nejnižšími náklady. Dále je také nutné zvolit vhodné materiály pro konstrukci tak, aby byl odolný a spolehlivý zároveň. Také je vytvořena dokumentace pro výrobu efektoru v podobě výkresu samotného efektoru (2.3) a jeho držáku (2.4).

## 2.2 Volba tvaru koncového efektoru

Při volbě typu konstrukce je nutné zaměřit se na specifikaci našich požadavků. Hlavním požadavkem je, aby zvolený efektor nadměrně nenamáhal dotykový displej a nedošlo tak k jeho poškození. Další velmi důležitou vlastností je, aby dotek efektoru dokázal vyvolat patřičný efekt u rezistivního i kapacitního displeje. Reakce u obou typů displejů musí být spolehlivě rozpoznatelná, aby nedocházelo k chybám vyhodnocení testů. Posledním z požadavků je, aby konstrukce zahrnovala prostor pro uchycení kamery, která bude snímat a vyhodnocovat činnost našeho efektoru.

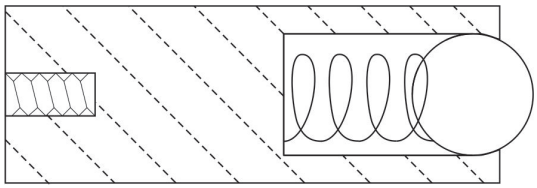
### 2.2.1 Samostatný stylus

Typů konstrukcí se nabízí hned několik. Za nejjednodušší konstrukcí by bylo možné považovat dotykový stylus, který by byl připevněn na konci robotického ramene. Teoreticky by byl takový postup možný, ovšem prakticky by s vysokou pravděpodobností došlo k poškození displeje nebo i celého zařízení. Bylo by nutné se plně spoléhat na dokonalou práci robotického ramene. Pokud si ale uvědomíme, jakou sílu takové rameno dokáže vyvinout, je zřejmé že při chybné definici pracovní roviny by k destrukci skutečně došlo. Je tedy zjevné, že toto řešení není tím, které je pro danou úlohu vhodné.

### 2.2.2 Efektor s kulovým mechanismem

Jinou možností by mohlo být sestrojit dotykový konec podobný kuličkové myši. Toto řešení je v porovnání s prvním konstrukčně, a tedy i finančně, výrazně nákladnější, přesto by ale přineslo jisté výhody. Jak je možné vidět na obrázku 2.1, byla by zde umístěna kulička v dutém válci, které by byla zevnitř tlačena pružinkou směrem ven

z dutiny. Zúžení tohoto válce na jeho konci zabraňuje jejímu vypadnutí ven. Výhodou je zde právě pružinka,která zabrání tvrdému nárazu do displeje. Pokud bude pracovní rovina manipulátoru špatně definována, tedy hlavně případ, kdy bude rovina doteku definována níže než skutečná rovina displeje, je pružinka tuto chybu schopna absorbovat a k poškození nedojde. Další velkou výhodou je, že s konstrukcí tohoto typu je robot schopen realizovat gesta, tedy tahy efektorem po displeji. Odvalování koule sníží tření a tím i opotřebení displeje. I tato konstrukce však má jisté nevýhody. Jednou z nich je právě koule. Kvůli zachovaní funkčnosti i pro kapacitní displeje musí být schopná vést náboj, respektive by měla být vodivá. [**?**] Nejlepším takovým vodičem je jistě kovová koule, ta by ale svým povrchem mohla způsobit poškození povrchu. Také pro vyvolání reakce na displeji je zapotřebí dotknout se dostatečně velkou plochou, aby byla vyvolána dostatečně velká změna kapacity (1.2.2). Řešením je najít kouli, která je vyrobena z pružné, ale zároveň vodivé hmoty. Dochází však ke zvýšení tření mezi koulí a vnitřní stěnou válce. Také pokud by měl být zákluz koule do pouzdra řádově v centimetrech, její průměr bude velmi narůstat, a výsledný efektor nabude obřích rozměrů. Důvod je zřejmý po pohledu na obrázek 2.1. Aby koule nevypadla z dutiny musí být hrdlo zúžené a tedy ven z pouzdra může vyčnívat vždy méně než polovina koule.



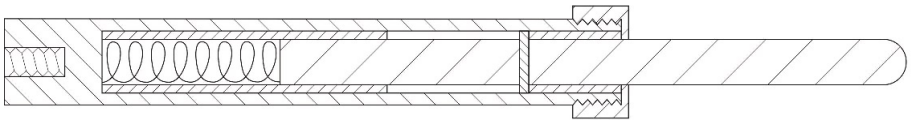
Obr. 2.1: Nákres řešení efektoru s koulí

### 2.2.3 Efektor se zákluzným mechanismem

Z důvodu uvedených nevýhod není vhodná ani koncepce s kuličkou. Možností je vytvořit koncepci spojením obou předchozích návrhů. Jako člen dotýkající se displeje je použit zakoupený stylus, který je právě pro tento účel vyvinut, a je pro něj vytvořen zákluzný mechanismus. Ten bude mít dvě využití. První jako ochrana před zničením vlivem chybné definice polohy ramene a druhé, jako přítlak stylusu pro lepší reakci u rezistivních displejů (1.2.1).

Pro stylus tedy je vytvořeno pouzdro, ve kterém se bude pohybovat stylus. Uvnitř pouzdra je tlačná pružina, která stylus tlačí směrem ven. Aby stylus neunikl z pouzdra, musí do něj být vložena závlačka. Ta se pohybuje uvnitř drážky, která je vytvořena v pouzdře. Zabraňuje vypadnutí stylusu, a zároveň také vymezuje rozchod

jeho pohybu. Pro zjednodušení uchycení efektoru k manipulátoru a zajištění proti vypadnutí závlačky je celé pouzdro vloženo do ještě jednoho obalu. Tím se zjednoduší i výroba vnitřního pouzdra na prostou trubičku s drážkami po bocích. Vnitřní pouzdro je zajištěno ve vnějším obalu převlečnou matkou. V zadní části pouzdra je vytvořena díra se závitem pro uchycení šroubem. Celá konstrukce je vyobrazena na obrázku 2.2.



Obr. 2.2: Nákres řešení efektoru se stylusem v zákluzném pouzdře

Je zjevné, že provést konstrukci s tímto zadáním lze více způsoby. Zvolená varianta je pouze jedna z mnoha, ale kvůli jejím vlastnostem je zvolena právě tato. Výhodnou vlastností je právě zákluzný mechanismus, který ochraňuje displej proti chybám manipulace až do třech centimetrů. Také konstrukce splňuje požadavek na použití pro kapacitní i rezistivní displeje. Nevýhodou této konstrukce by mohlo být to, že je příliš masivní, ale jelikož je jedním z požadavků i uchycení kamery a je zapotřebí, aby stylus byl delší než kamera a tím ji chránil před poškozením, je jeho robustnost výhodou.

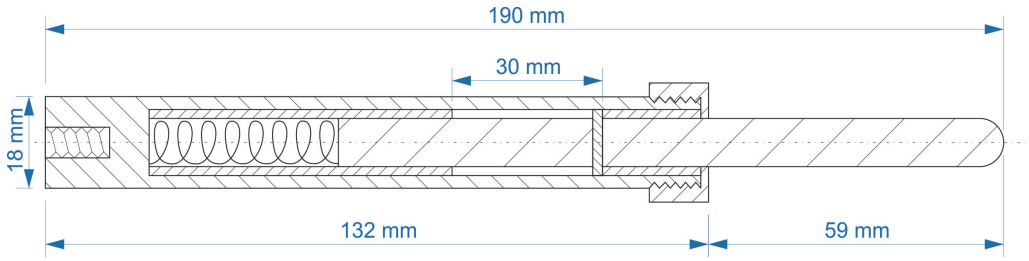
Posledním prvkem je část pro uchycení kamery, která zároveň spojí efektor a rameno. Je vytvořen z plechu tloušťky 1,5 mm ve tvaru písmene L a jsou do něj vyvrtány otvory pro uchycení kamery, pro uchycení efektoru a také pro upevnění celé konstrukce k manipulátoru. Držák je na obrázku 2.4.

## 2.3 Volba materiálu pro výrobu

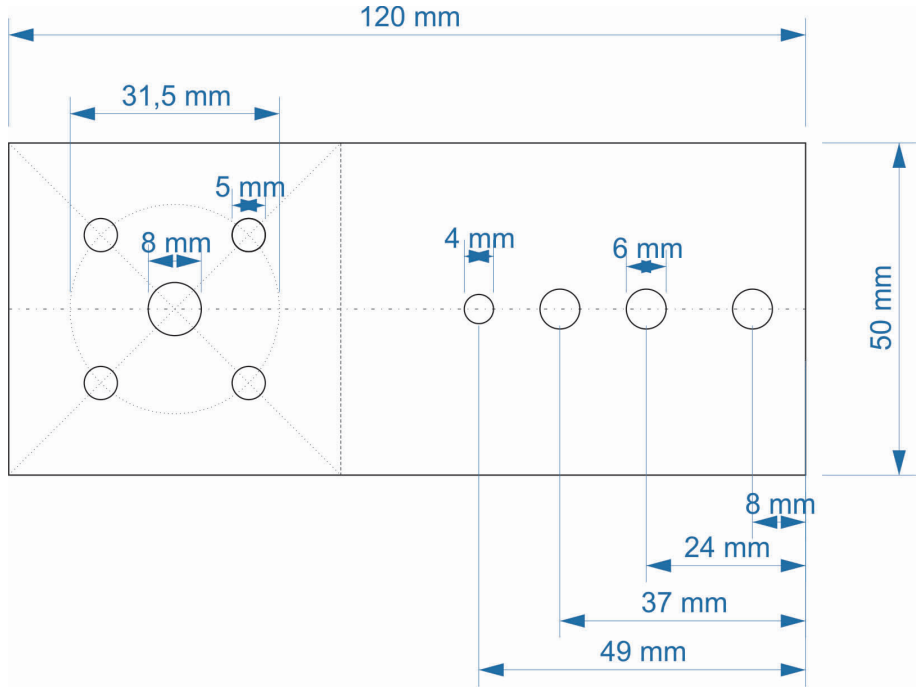
Při volbě materiálu je důležité, aby zvolený materiál byl dobře opracovatelný a pevný. V první fázi výběru materiálů bylo zvoleno železo na vnější plášť a teflon na vnitřní trubičku. Teflon měl být použit kvůli jeho nízkému součiniteli tření. [1] Nakonec jsou zvoleny jako materiály bronz a měď. Byl zajištěn váleček bronzu měděná trubička. Výhodou těchto kovů je, že nepotřebují po opracování žádnou další povrchovou úpravu.[2] Bronz byl použit pro vnější pouzdro, především kvůli jeho lepším mechanickým vlastnostem, je pevnější a opracovává se lépe než měď. Pro vnitřní pouzdro je použita měď a jako závlačka malý měděný váleček. Pro uchycení kamery plech potažený zinkem o tloušťce 1,5 milimetru, aby byl dostatečně pevný. Zinek slouží jako ochranná vrstva železa proti korozi. Stylus je výrobkem firmy Genius a nese označení Touch Pen 80S.

## 2.4 Výroba efektoru

Pro výrobu je vytvořena patřičná dokumentace, tedy výkres ze kterého je možné vyčíst, který prvek má mít jaké rozměry a jak bude uložen. Výkresy jsou nakresleny v měřítku 1:1 pro efektor i pro jeho uchycení. Tyto výkresy jsou na obrázcích 2.3 a 2.4.



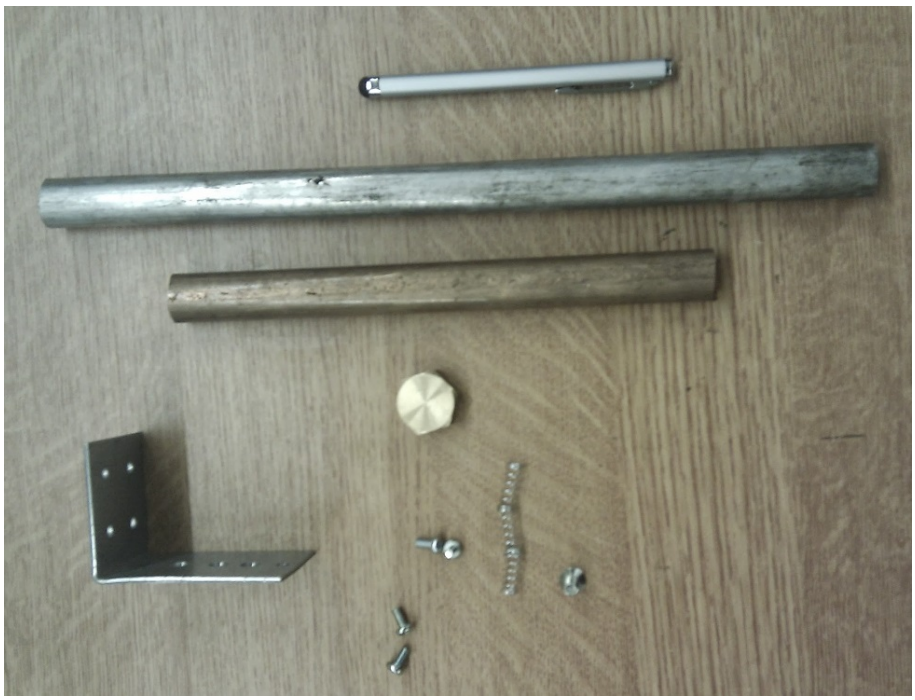

Obr. 2.3: Výrobní výkres efektoru



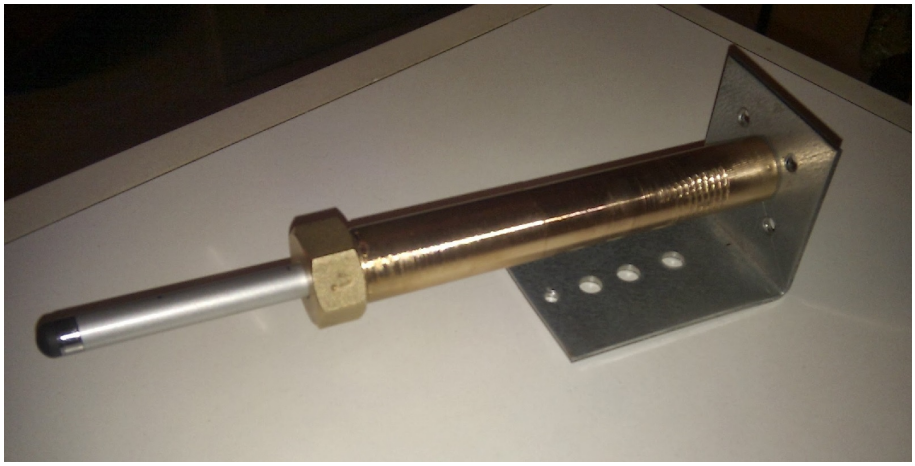

Obr. 2.4: Výrobní výkres uchycení kamery a efektoru

První vyrobenou částí je vnější pouzdro. Bronzový váleček bylo proto nutné nejprve zkrátil na požadovanou délku a vysoustružil na správný průměr. Poté bylo třeba odvrtat vnitřek válce pro vložení vnitřního pouzdra a také z opačné strany pro přichycení. Do pouzdra jsou vyřezány závity pro převlečnou matku i pro šroub

uchyceni. Z mědi byl vysoustružen dutý váleček a jelikož nebyla k dispozici frézka, drážky jsou vytvořeny za pomoci navrtání děr a následného jejich vzájemnému propojení pilníkem. Poslední součástí je držák kamery. Byl ustřižen plech správného formátu, dle výkresu navrtány díry, následně díry a hrany obroušeny a celý plech ohnut do pravého úhlu. Nakonec byl provrtán stylus a všechny komponenty seskládány. Materiál pro výrobu efektoru je na fotografii 2.5 a výsledná sestava je na fotografii 2.6. Na fotografii s materiálem pro výrobu chybí měděná trubička, naopak je zde navíc hliníkový váleček. To z důvodu, že v prvotní fázi výroby byl pro vnitřní pouzdro použit právě hliník, ale opracování jeho povrchu tak, aby byl hladký a stylus po něm snadno klouzal, bylo v dostupných podmínkách nemožné. Proto v průběhu výroby byl změněn materiál za měděnou trubičku, která byla dostupná v požadovaných rozměrech a nebylo zapotřebí opracovávat její povrch. Robot s instalovaným efektorem včetně držáku kamery je na obrázcích 2.7 a 2.8.



Obr. 2.5: Materiál pro výrobu efektoru



Obr. 2.6: Dokončená konstrukce efektoru


Obr. 2.7: Robot s efektorem


Obr. 2.8: Robot s efektorem a kamerou

# 3 OBSLUŽNÝ SOFTWARE

## 3.1 Rozdělení programu

Tato části práce se zabývá programem pro ovládání robota. Nejde o popis způsobu samotného řízení robota, pro ten je vytvořeno komunikační rozhraní, které převádí příkazy z ovládací třídy (driveru v jazyce C#) do zpráv , které řídící jednotka robotu programovaná v jazyce SPEL+ rozklíčuje a vykoná. Detaily ovládací třídy a způsobu komunikace jsou popsány v práci Obslužný program pro robotický manipulátor epson [6]. Tato kapitola se zabývá popisem obslužného softwaru. Celý program je napsán v jazyce C#.

V první části jsou popsány tři třídy. Třída RobotClicker (3.2.1), která funguje jako prostředník v komunikaci s ovladačem manipulátoru. Dále třída Test_IO (3.3.1), která zajišťuje ukládání a vyčítání jednotlivých testů a jejich nastavení a třídy Settings (3.4), která spravuje obecné nastavení aplikace.

Ve druhé části kapitoly je popsána ovládací aplikace pro windows, její možnosti, funkce, způsob nastavení a ovládání. Na konci kapitoly je obrázek celé sestavy při testování programu (3.17).

## 3.2 Ovládací rozhraní robotu

Celá tato třída je vytvořena jako kompakt obsahující potřebné funkce a proto je její využití snadné. Stačí ji pouze vložit do vlastního projektu společně s třídami ovladačů robotu od studenta Fireše a poté už jen využívat metody. Třída také v metodách rozlišuje, zda se nachází v normálním režimu nebo režimu simulace. Je tedy schopna vracet hodnoty a vypisovat souřadnice pohybu i bez připojení k robotu.

### 3.2.1 Popis metod ovládacího rozhraní

#### Konstruktory třídy RobotClicker

*public* RobotClicker()
*public* RobotClicker( *string* _name )

Konstruktor bez parametru vytvoří třídu RobotClicker se základním nastavením všech úhlu koncového bodu a definicí výchozí (domácí) pozice. Také konstanty homogenní transformace zůstanou v základním tvaru, kdy ke transformaci nedochází. Navíc v tomto výchozím nastavení není známo, v jaké poloze se nachází displej, a

proto jsou i hodnoty pro dotek nastaveny do úrovně výchozí pozice, aby nemohlo dojít k poškození zařízení.

Pokud ale pro vytvoření třídy bude použit konstruktor s parametrem typu *string*, kde parametrem je jméno testu, načtou a nastaví se všechny proměnné dle uložených hodnot pro daný test a robot je poté připraven na úpravu či spuštění testu daného jména.

### Metoda Connect

*public Conn* Connect(*string* IPAdr, *string* Port)

Metoda Connect je veřejná a má dva vstupní parametry. Oba jsou datového typu *string*. První z nich je hodnota IP adresy robota a druhý parametr portu, ke kterému se má připojit. Metoda má návratovou hodnotu typu výčet (*enum*). Hodnota, kterou vrací, závisí na úspěšnosti připojení nebo chybě, která se při připojování vyskytla. Pokud je spojení úspěšně navázáno návratová hodnota bude 1 (connect), v opačném případě bude 0 (fail). Bude-li chyba ve vstupech, vrátí hodnotu 4 (formatexcp) pro chybu IP adresy, nebo hodnotu 3 (socketexcp) pro chybu soketu. Pokud se podaří připojit k robotu, odešle se na něj inicializace, nastaví se výchozí nástroj metodou UseTool (3.2.1) a odešle se robot na výchozí pozici pomocí metody GoHomePosition (3.2.1). Metoda UseTool je volána proto, aby byl pracovní nástroj jasně definován a nestalo se, že se pracuje s nástrojem, který používala nějaká předchozí aplikace.

### Metoda Disconnect

*public bool* Disconect()

Metoda Disconnect nemá žádný vstupní parametr. Slouží pouze pro odpojeni od robota a smazání komunikačního objektu, který byl vytvořen při připojování. Návratová hodnota typu bool oznamuje, zda odpojení proběhlo a pokud ano, vrátí true. Pokud ale robot nebyl připojen a metoda bude volána, neprovede se nic a návratová hodnota bude false.

### Metoda IsConnected

public bool IsConnected()

Tato metoda vrací hodnotu true, pokud je robot v daný okamžik připojen a false, pokud připojen není. Metoda je určena především pro použití v podmínkách ochran programu.

#### Metoda UseTool

public void UseTool()

Metoda UseTool() pouze volá metodu SendToolSet třídy ovladače a nastavuje hodnoty všech souřadnic na nulu.

#### Metoda GoHomePosition

*public void* GoHomePosition()

Metoda GoHomePosition je opět o veřejnou metodu bez parametru. I když je to samostatná metoda, uvnitř pouze předá souřadnice domácí pozice metodě PerformClickLoader(HomePosX , HomePosY, HomePosZ) a zavolá metodu PerformClick (3.2.1), která zajišťuje odesílání dat robotu.

#### Metoda PerformClickLoader

*public bool* PerformClickLoader(*List<string>* inList)
*public void* PerformClickLoader(*List<UserControl>* UCList)
*public void* PerformClickLoader(*UserControl* UC)
*public void* PerformClickLoader(*double* X, *double* Y)

První z těchto metod je významnější tím, že z každého bodu o souřadnicích X a Y, který je do ní poslán vytvoří tři body. Souřadnice X a Y se nemění, ale mění se souřadnice Z. První hodnota Z odpovídá rovině nad displejem, druhá je v úrovni doteku a třetí opět v rovině nad displejem. Všechny tyto vytvořené body přidá do *List<string>* inListHolder a jako poslední bod přidá souřadnice výchozí pozice. Poté volá metodu PerformClick (3.2.1), která zajišťuje odeslání dat robotu. Všechny ostatní metody, tedy druhá, třetí a čtvrtá pouze transformují souřadnice ze vstupu do podoby vhodné pro první z metod PerformClickLoader a poté jí předají data.

#### Metoda PerformClickLoaderNoTransform

*public void* PerformClickLoaderNoTransform(*double* _X, *double* _Y, *double* _Z, *double* _U, *double* _V, *double* _W)

Tato metoda je specifická tím, že souřadnice jsou přímo odeslány na robota bez použití homogenní transformace. Právě z tohoto důvodu jsou vyžadovány všechny hodnoty definice bodu v prostoru, i jeho natočení. Metoda je v programu využívána pro definici počáteční pozice robotu.

#### Metoda PerformClick

*private bool* PerformClick(*bool* Transform = true)

Metoda PerformClick má nepovinný parametr a je privátní. Volání bez parametru provádí transformaci souřadnic, zatímco volání s parametrem false odešle souřadnice přímo. Vždy rozebere první data typu *string* z globální proměnné *List<string>* inListHolder a separuje hodnoty X, Y a Z, které uloží do globálních proměnných Xval, Yval a Zval. První řetězec globálního seznamu je poté smazán a z uložených souřadnic je vytvořen nový objekt třídy RobCoord a ten je odeslán robotu, který poté najede na požadovaná souřadnice. Uvnitř této metody, při volání bez parametru, probíhá homogenní transformace (1.4), která body transformuje podle úhlového natočení celého systému a jeho posunu od výchozích souřadnic.
Při vyvolání této metody však nedojde jen k nastavení robotu do polohy prvního bodu, ale postupně si automaticky vyčítá další body a najíždí do jejich poloh. Posun na další bod však není aktivován uvnitř metody, ale je spouštěn zvenčí metodou DataRead (3.2.1).

#### Metoda DataRead

*private void* DataRead()

Tato metoda je obslužnou metodou pro událost (event), kterou generuje třída ovladače robotu. Metoda vyčte parametry události a zjistí, zda se jedná opravdu o dokončení pohybu robota a pokud ano, zjistí, která souřadnice byla právě dosažena. Pokud se jedná o spodní pozici, vygeneruje se událost o doteku (ClickDone). Bude-li dosažená pozice zároveň poslední ze seznamu souřadnic, vygeneruje se událost o dokončení všech pohybů (ClickDoneAll).

## 3.3 Ukládání dat testů

### 3.3.1 Popis metod ukládání dat testů

#### Konstruktor Test_IO

*public* Test_IO()

Při vytvoření třídy se konstruktor pokusí o načtení seznamu testů ze souboru Test_list.xml do vnitřní globální proměnné File_xdoc typu *XDocument*. Pokud se načtení nepodaří, vytvoří nový prázdný soubor seznamu testů.

### Metoda Save_TestSetting

*public void* Save_TestSetting(*string* _name , *XDocument* _set_xdoc)

Metoda Save_TestSetting dostává na vstupu jméno testu a data nastaveni testu v proměnné typu *XDocument*. Při volání si metoda načte a aktuální verzi souboru Test_list.xml, uvnitř dokumentu vytvoří nový element testu. V tomto elementu jsou vytvořeny další dva elementy, přičemž jeden obsahuje název testu ze vstupu a do druhého je vložena celá struktura dat nastavení. Poté je soubor opět uložen pod jménem Test_list.xml.

### Metoda Save_Test

*public void* Save_Test(*string* _name, *List<UserControl>* Points)

Metoda Save_Test má jako vstupní parametry jméno testu typu *string* a seznam bodů ve formátu *List<UserControl>*. V seznamu jsou uloženy dva typy bodů, body(Point) a kontrolní body (Checkpoint). Oba z těchto typů mají některé odlišné vlastnosti.

Při ukládání se nejdříve vyprázdní proměnná Test_xdoc typu *XDocument*, aby nedošlo k tomu, že do definovaného testu budou nechtěně přidány body, které do něj nepatří. Poté se vytvoří základní element dokumentu, který také nese jméno testu. Jednotlivé body a kontrolní body jsou v cyklu vysázeny do nosného elementu. Každý z vložených bodů je elementem, který se jmenuje "Point"a obsahuje další elementy s jednotlivými parametry každého z bodů, přičemž první z nich nese informaci o typu bodu. Po vysázení všech bodů do dokumentu je uložen jako Jméno_testu.xml.

### Metoda Load_TestList

*public List<string>* Load_TestList()

Metoda nejdříve vymaže data z proměnné list typu *List<string>*, načte do proměnné File_xdoc soubor Test_list.xml a v cyklu přidá všechna jména testů ze souboru do seznamu list, který předá jako návratovou hodnotu. Když je metoda volána při prvním spuštění programu, je možné, že soubor Test_list.xml ještě neexistuje. V takovém případě metoda vrátí hodnotu null.

### Metoda Del_test

*public void* Del_test(*string* _name)

Metoda Del_test je určena pro úplné odstranění testu. Nejdříve se smažou všechny uložené fotografie pomocí metody Del_photos (3.3.1). Dále se načte aktuální verze souboru Test_list.xml, vyhledá se záznam testu příslušného jména. Záznam se odstraní a změny uloží do souboru. Nakonec se vymaže soubor definice testu Jméno_testu.xml a konfigurační soubor kamery pro daný test Jméno_testu.camset.

#### Metoda Del_photos

*private void* Del_photos(*string* _name)

Metoda je určena pro smazání všech obrázků testu zvoleného jména. Samostatnou metodou je především z důvodu zvýšení přehlednosti programu. Uvnitř pracuje tak, že si načte data souboru definice testu a postupně z elementů všech bodů čte jména souborů obrázků a ty smaže.

#### Metoda Load_Test_InList

*public List<UserControl>* Load_Test_InList(*string* _name)

Metoda pro načtení všech bodů testu do proměnné Points_IO typu *List<UserControl>*. Seznam Points_IO se nejdříve vyprázdní a poté se do něj v cyklu přidávají všechny body z definičního souboru testu Jméno_testu.xml. Když jsou ze souboru vyčteny všechny hodnoty je seznam předán jako návratová hodnota metody.

#### Metoda Rename_test

*public void* Rename_test(*string* _name, *string* _newname)

Metoda změní jméno testu z původního na nové. Obě jména jsou vstupními parametry typu *string*. V prvním kroku se načte soubor seznamu testů Test_list.xml, najde se test původního jména, změní se na nové a soubor se uloží. Ve druhém kroku se přejmenuje soubor definice testu a soubor nastavení kamery. A ve třetím kroku se postupně projde definiční soubor testu a jména uložených obrazů a odkazy na ně se starými jmény jsou nahrazeny novými.

#### Metoda Get_Test_setting

*public XDocument* Get_Test_setting (*string* _testname)

Tato metoda načte soubor seznamu testů Test_list.xml a vyčte z něj hodnoty nastavení pro test jména ze vstupního parametru. Tyto hodnoty nastavení vloží do proměnné typu *XDocument* a předá jako návratovou hodnotu metody.

## 3.4 Ukládání nastavení aplikace

Tato třída byla poskytnuta vedoucím práce Ing. Adamem Chromým a slouží pro k ukládání nastavení aplikace do souboru *App.config*. Jednoduchým způsobem umožňuje ukládat a vyčítat hodnoty IP adres kamery a robotu a portu robotu. Uložení nastavení vypadá takto *settings ["IPcam"] = "10.0.41.10";* , naopak vyčtení ze souboru *promenna = settings["IPcam"];*.

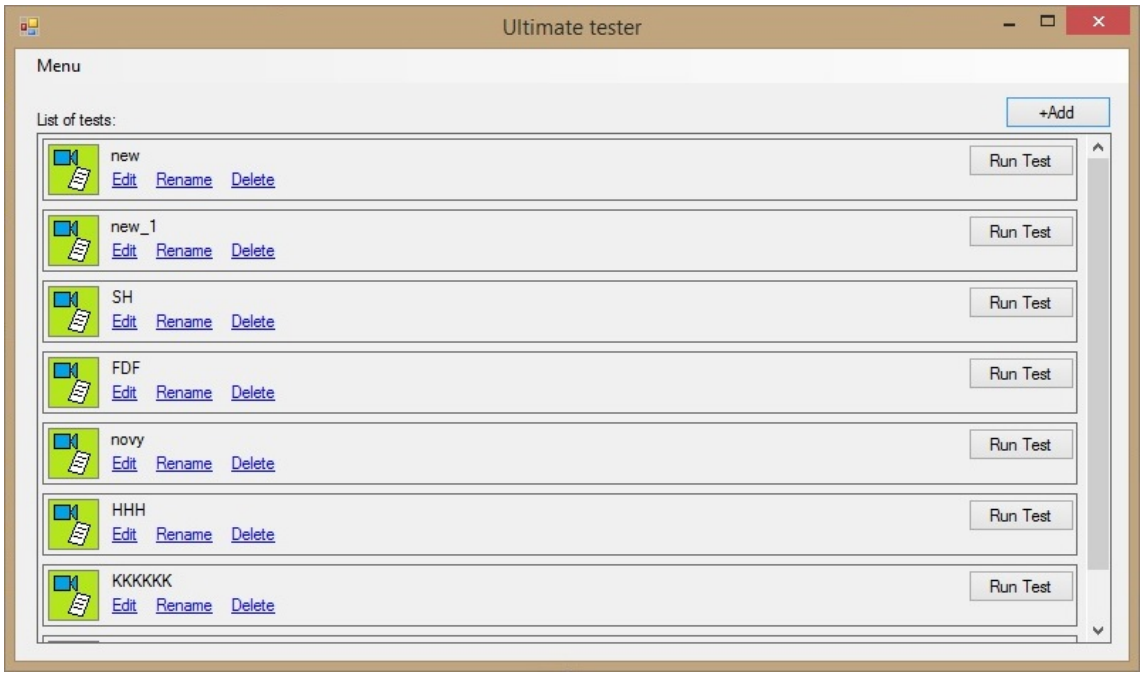
## 3.5 Popis ovládání aplikace

V následujících podkapitolách jsou popsána jednotlivá okna aplikace, se kterými se setkáme v průběhu jejího ovládání. Jsou zde popsána okna pro nastavení a definici testů, okna pro jejich spouštění, jejich ovládání a také metody, které zajišťují správnost funkce celého programu. Je nutné také zmínit, že do aplikace je vložena třída ovladače kamery, díky které je možné vyčítat její obrazy, ukládat je a měnit nastavení kamery. Instalační balíček ovladače kamery je obsažen v digitální formě práce v příloze na CD disku. Aby bylo možno ovládací třídu použít, je zapotřebí po nainstalování přidat ve Visual Studiu referenci na tento ovladač.

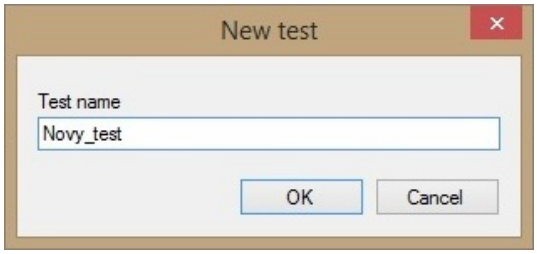
### 3.5.1 Hlavní okno aplikace

Form_Main (obr. 3.1) je první okno (form), které se zobrazí, po spuštění aplikace. Toto okno samo o sobě nabízí dvě možnosti. První z nich je nahoře v záložce Menu položka nastavení (Settings). Po stisknutí této záložky se otevře okno s obecným nastavením aplikace (obr. 3.4). Druhou možností je vytvoření nového testu stisknutím tlačítka +Add. Vyskočí okno pro zadání jména nového testu (obr. 3.2), kam se název vepíše a stisknutím tlačítka OK se jméno testu potvrdí. Po potvrzení názvu se objeví okno pro výchozí nastavení nového testu (3.5.5). Další možnosti, které jsou v hlavním okně přístupné, jsou ovládací prvky jednotlivých, již vytvořených testů, které jsou vyobrazeny jako UserControl (obr. 3.3), který má každý jeden test svůj vlastní. V panelu testu jsou obsaženy možnosti pro úpravu testu (3.5.2), přejmenování (3.5.2), smazání (3.5.2) a spuštění (3.5.2). Úprava testu (edit) otevře okno pro definici testu (3.5.7), ve kterém je možné test upravit. Možnosti přejmenování

(rename) a smazání testu (delete) volají funkce z třídy hlavního okna. Poslední možností panelu je spuštění testu, které otevře nové okno pro přípravu spuštění testu (3.5.12). Detaily funkcí hlavního okna jsou popsány v kapitole 3.5.2 a UserControlu hlavního okna v kapitole 3.5.3.




Obr. 3.1: Hlavní okno ovládací aplikace (3.5.1)




Obr. 3.2: Okno zadání jména nového testu (3.5.2)


Obr. 3.3: Panel zobrazení testu (3.5.3)

### 3.5.2 Hlavní okno aplikace - popis metod

#### Konstruktor třídy Form_Main

*public* Form_Main()

Konstruktor třídy Form_Main po inicializaci komponent načte seznam testů ze souboru (3.3.1) a postupně vytvoří panely (UserControl_Main 3.5.3) všech testů, nastaví metody pro obsluhu jejich událostí (eventů) a přidá je do FlowLayoutPanelu hlavního okna.

#### Metoda Form_Main_Play_me

*private void* Form_Main_Play_me(*object* sender, *EventArgs* e)

Metoda Form_Main_Play_me je obsluhou události (eventu), která je generována v panelu testu (obr. 3.3) po stisknutí tlačítka Run test. Metoda otevře nové okno pro natavení spuštění testu (obr. 3.14) a po jeho potvrzení je spuštěno okno samotného testu (obr. 3.15).

#### Metoda Form_Main_Rename_me

*private void* Form_Main_Rename_me(*object* sender, *EventArgs* e)

Metoda je aktivována událostí, vyvolanou kliknutím na text Rename v panelu testu (obr. 3.3). Při jejím spuštění se otevře okno pro zadání nového jména testu (3.5.2). Po jeho potvrzení se nejdříve zkontroluje, zda jméno není shodné s jiným uloženým testem. Pokud je možné nové jméno použít, je volána metoda Rename_test (3.3.1), změněno jméno na panelu testu a ve vnitřním seznamu je odebráno staré jméno testu a přidáno nové.

#### Metoda Form_Main_Edit_me

*private void* Form_Main_Edit_me(*object* sender, *EventArgs* e)

Metoda je spouštěna událostí textu Edit v panelu testu (obr. 3.3). Aktivací se otevře okno (obr. 3.10), ve kterém je možno test libovolně upravovat. Po dokončení úprav se kliknutím na tlačítko Save změny uloží. Pokud je ale okno uzavřeno křížkem a nebo pomocí tlačítka Cancel, žádná ze změn není uložena a data která byla vytvořena během úprav jsou zmazána.

#### Metoda Form_Main_Delete_me

*private void* Form_Main_Delete_me(*object* sender, *EventArgs* e)

Metoda je, stejně jako obě předchozí, obsluhou události generované panelem testu. Po kliknutí na text Delete nejdříve vyskočí okno s upozorněním, zda opravdu chceme daný test smazat. Potvrzením tohoto okna je aktivována metoda Del_test (3.3.1), která smaže všechna data testu a poté je odebrán i panel testu z hlavního okna.

#### Metoda button_Add_Click

*private void* button_Add_Click(*object* sender, *EventArgs* e)

Metoda je spuštěna zmáčknutím tlačítka +Add. Nejdříve se otevře okno pro zadání jména nového testu (obr. 3.2). Pokud je toto jméno možné použít, spustí se okno nastavení testu (obr. 3.5). Okno uživatele naviguje a po dokončení nastavení se spouští okno definice samotného testu (obr. 3.10). Potvrzením tohoto okna se uloží veškeré soubory definice a nastavení testu a v hlavním okně (obr. 3.1) se vytvoří nový panel testu (obr. 3.3). Pokud si uživatel vytváření testu ve kterémkoli z kroků rozmyslí a okno uzavře, všechna data jsou okamžitě smazána.

#### Metoda InputBox

*public static DialogResult* InputBox(*string* title, *string* promptText, *ref string* value)

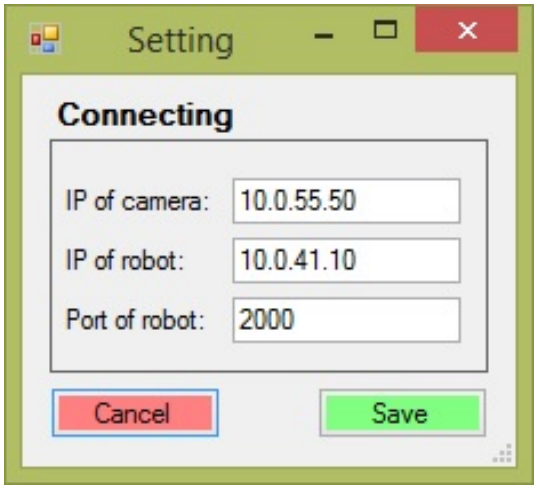
Tato metoda byla převzata od vedoucího práce Ing. Adama Chromého. Metoda vytváří okno, do kterého je možné vepsat text a ten následně předat jiné metodě či třídě. Vstupní parametry mají následující význam. Title - titulek otevřeného okna, promptText - text vepsaný v okně po spuštění a value je hodnota výstupu, kterou je po potvrzení textu v okně možné předat dál. Toto okno je na obrázku 3.2.

### 3.5.3 Panel testu - popis metod

UserControl_Main je na obrázku 3.3, obsahuje čtyři ovládací prvky a má jednu vlastnost (property). Vlastností je jméno testu, tedy je přímo vázána na popisek (label). Ovládacími prvky jsou jsou tři nápisy typu *linkLabel* a jedno tlačítko (button). Všechny ale mají totožnou funkci a to, že po jejich aktivaci vygenerují událost (event), která je zpracovávána metodami v jiných třídách.

### 3.5.4 Okno obecného nastavení aplikace

V okně nastavení jsou tři kolonky pro na stavení hodnot a dvě tlačítka, jedno pro uložení změn a druhé pro jejich zrušení. V jednotlivých kolonkách je možné změnit IP adresu pro připojení manipulátoru, port připojení a IP adresu pro připojení kamery. Tato metoda pro své funkce využívá třídu Settings (3.4). Okno je na obrázku 3.4.

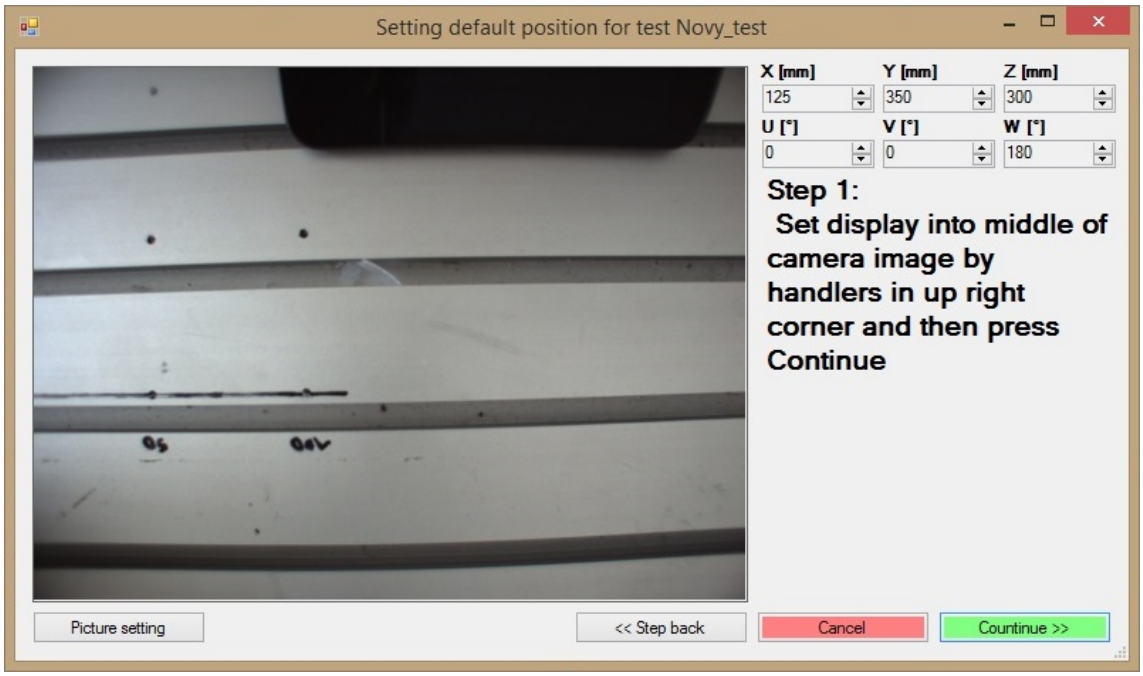



Obr. 3.4: Okno hlavního nastavení aplikace (3.5.4)

### 3.5.5 Okno vytvoření nového testu

Při nastavování nového testu se otevře okno, které je na obrázku 3.5. Celé nastavení se skládá ze čtyř kroků a je umožněno se mezi nimi volně pohybovat tam i zpět pomocí tlačítek *Step back* a *Continue* Také je zde tlačítko Cancel, kterým se celá definice nového testu ukončí. Posledním z tlačítek je Picture setting, které vyvolá okno nastavení obrazu (obr. 3.9) ze třídy ovladače kamery. Toto nastavení je možné vyvolat kdykoliv během nastavování a změnit jej. Největším prvkem okna je rámec, ve kterém je spuštěn přímý přenos obrazu z kamery. V pravém horním rohu jsou prvky pro nastavení polohy robotu. Pomocí nastavení U, V a W je otáčeno rovinou pohybu robotu. Změnou X, Y a Z je pohybováno robotem v daných osách, ale otočených podle nastavení roviny. Pod těmito nastavovacími prvky je zobrazen text, který uživatele naviguje v obsluze programu.

Krok 1
V prvním kroku (obr. 3.5) se pomocí ovládacích prvku nastaví robot do takové polohy, aby byl efektor namířen kolmo proti displeji a zároveň se displej nachází uprostřed obrazu kamery. Přechod do dalšího kroku se potvrdí tlačítkem *Continue*.




Obr. 3.5: Zadávání výchozí pozice robotu krok 1 (3.5.5)

Krok 2

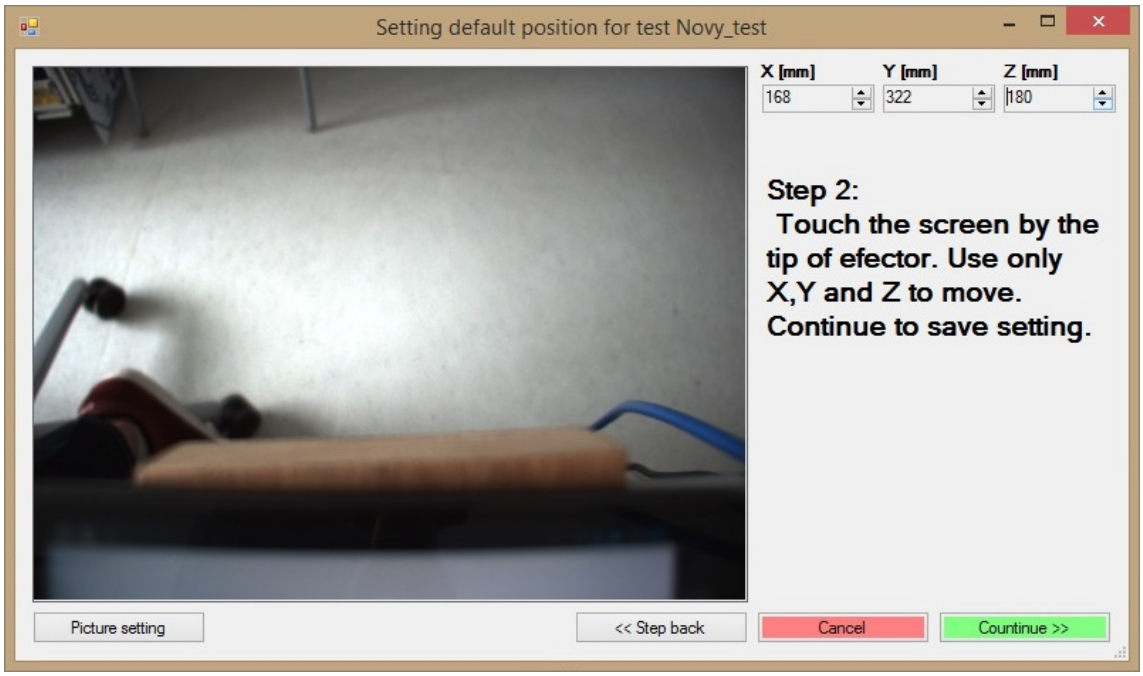
Ve druhém kroku (obr. 3.6) už nelze měnit natočení manipulační roviny. V této chvíli musí obsluha pomocí ovládacích prvků X, Y a Z nastavit tak, aby se špička efektoru dotkla displeje. Krok je nutno opět potvrdit tlačítkem *Continue*.

Krok 3

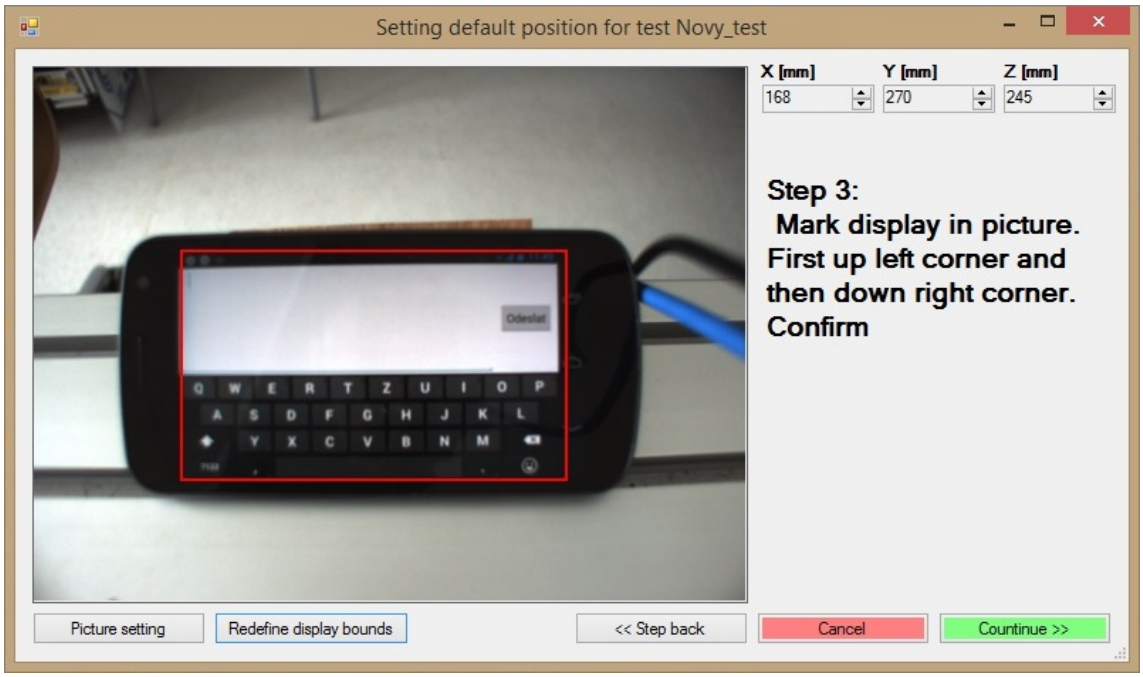
Přechodem do kroku tři (obr. 3.7) se robot vrátí do své výchozí polohy (nastavená v kroku 1). Nyní musí operátor označit displej v obrazu. Provede tak kliknutím myši nejdříve do levého horního, a poté do pravého spodního rohu. V případě, že vykreslený obdélník neodpovídá okrajům displeje v obrazu, je možné pomocí tlačítka *Redefine display bounds* nastavení smazat a označit displej znovu. Když je displej označen, je možné přejít k dalšímu kroku.

Krok 4

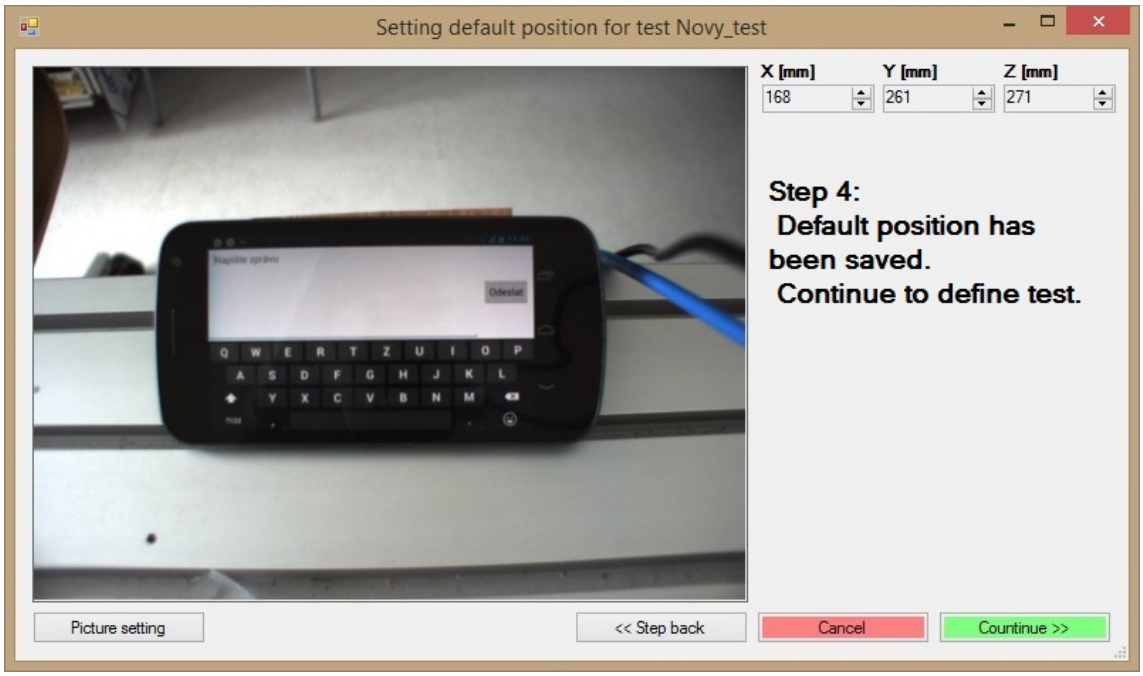
Přechodem do posledního kroku (obr. 3.8) je potvrzeno nastavení celého testu. Před jeho uložením ještě proběhnou korekce. Především se robot přesune tak, aby byl přesně nad středem displeje, což je možné díky označení displeje v obrazu. Potvrzením posledního kroku se aktuální okno uzavře a otevře se okno definice testu (obr. 3.10).



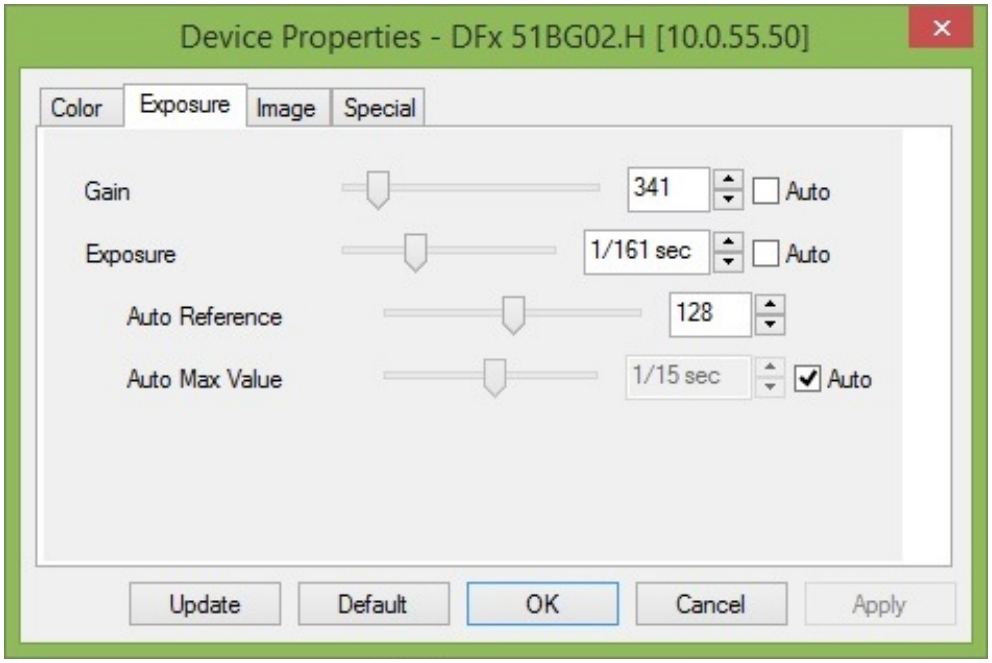

Obr. 3.6: Zadávání výchozí pozice robotu krok 2 (3.5.5)




Obr. 3.7: Zadávání výchozí pozice robotu krok 3 (3.5.5)




Obr. 3.8: Zadávání výchozí pozice robotu krok 4 (3.5.5)




Obr. 3.9: Okno nastavení parametrů snímání kamery (3.5.5)

### 3.5.6 Okno vytvoření nového testu - popis metod

**Konstruktor Form_HomePos**

*public* Form_HomePos()

V konstruktoru jsou implementovány metody pro spuštění přenosu obrazu z kamery a také pro připojení k robotu. Pokud se některé z těchto připojení nezdaří, je na to operátor upozorněn vyskakovacím oknem. Po připojení ke kameře je u ní nastaven formát obrazu a je spuštěn časovač pro obnovování obrazu (3.5.6). Také je spuštěn časovač pro odesílání dat do robotu (3.5.6)

### Metoda timer1_Tick

*private void* timer1_Tick(*object* sender, *EventArgs* e)

Metoda obsluhuje časovač a je tedy pravidelně volána. V každém volání metoda načte obraz ze třídy kamery a poté tento obraz vykreslí do pictureBoxu. Navíc, pokud je splněna podmínka, že krok je roven třem a souřadnice vykreslovaného obdélníku jsou různé od nuly, přepočtou se souřadnice a obdélník je vykreslen do obrazu. Přepočet souřadnic probíhá kvůli možnosti změnit velikost okna během nastavování a je nutné, aby obdélník byl vykreslen stále ve stejné pozici v obrazu.

### Metoda timer2_Tick

*private void* timer2_Tick(*object* sender, *EventArgs* e)

Tato metoda obsluhuje druhý časovač. Při její aktivaci se z prvků nastavení natočení roviny vyčtou hodnoty a uloží do proměnných. Poté se vypočtou koeficienty homogenní transformace (1.4) pro dané natočení systému a vypočten posun robota ve všech třech rovinách od vztažného bodu. Vypočtou se transformované souřadnice a tyto se předají robotu pomocí metody PerformClickLoaderNoTransform (3.2.1), aby transformace neprobíhala znovu.

### Metoda Setting_step

*private void* Setting_step(*int* _step)

Tato metoda slouží k přepínání kroků nastavování testu. Je volána stiskem tlačítka *Step back* a *Continue*. Jako parametr dostává číslo typu *int* s hodnotou kroku, do kterého se má okno přenastavit. Uvnitř metody je přepínač (switch) a ten vybere příslušnou část kódu pro nastavení. Nastavení jednotlivých kroků je zde popsáno. Některé nastavení v jednotlivých krocích se mohou zdát zbytečná, ale jsou zde kvůli možnosti vracet se v průběhu nastavování.

V prvním kroku se nastaví viditelnost ovládacích prvků natočení pracovní roviny robotu jako viditelné a konstanty homogenní transformace se nastaví tak, aby k transformaci nedocházelo. Také se nastaví text průvodce nastavením na text kroku jedna.

Po přechodu do kroku dvě se nastaví text průvodce na popis druhého kroku. Uloží se hodnoty výchozí pozice robotu a viditelnost ovládacích prvků natočení roviny se nastaví jako neviditelné. Pokud je robot odpojen, pokusí se opět připojit. Tlačítko *Redefine display bounds* se nastaví také jako neviditelné.

Ve třetím kroku se nastaví text průvodce, odečte se hodnota souřadnice Z (výška displeje) a tlačítko *Redefine display bounds* se nastaví jako viditelné. Díky tomu, že je známa výchozí pozice, výška displeje a jsou známy i pozorovací úhly kamery, lze určit meze souřadnic, které je možné v obrazu vidět. Do ovládacích prvků jsou zapsány hodnoty výchozí pozice a jsou vynulovány proměnné souřadnic obdélníku.

V posledním kroku je vypočten posun displeje v obrazu jako rozdíl středu obrazu a středu vyznačeného obdélníku. Jsou provedeny korekce souřadnic a změny zapsány do ovládacích prvků. Robot najede na korigovanou pozici a odpojí se. Tlačítko *Redefine display bounds* je nastaveno jako neviditelné. Z celého definovaného nastavení je vytvořena xml struktura, která je uložena později po definici testu pomocí metody Save_TestSetting (3.3.1). Nastavení kamery je ale uloženo přímo jako Jméno_testu.camset

#### pictureBox1_Click

*private void* pictureBox1_Click(*object* sender, *EventArgs* e)

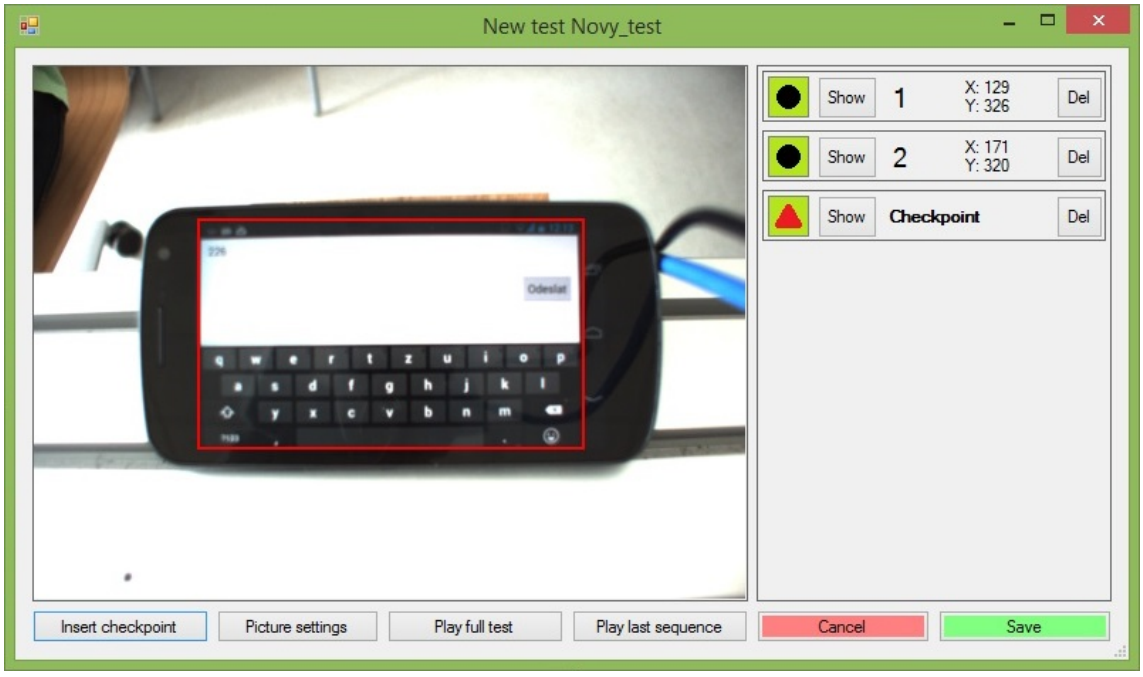
Metoda je aktivní pouze pokud se nacházíme v kroku 3 a alespoň jedna ze souřadnic pro vykreslení rámečku je nulová a bod kliknutí je přímo v obrazu. Při roztažení okna se totiž stane, že kolem obrazu jsou šedé pruhy, ve kterých metoda na kliknutí reagovat nesmí. Pokud jsou tedy splněny všechny podmínky, dojde k přepočtu pozice kliknutí a je uložena při prvním kliknutí jedna a při druhém druhá souřadnice obdélníku. K přepočtu musí docházet právě kvůli možnosti roztažení okna. Nesmí se totiž stát, aby bylo možné displej označit mimo obraz kamery.

### 3.5.7 Okno definice nového testu

V okně definice nového testu (obr. 3.10) je opět dominantní panel, ve kterém se promítá obraz z kamery. Napravo od tohoto je panel, ve kterém se zobrazují jednotlivé zvolené body. Na spodní hraně okna se nachází šest tlačítek. Tlačítko *Insert checkpoint* slouží pro vložení kontrolního bodu testu (obr. 3.12). Lze je ale zmáčknout jen

tehdy, pokud je manipulátor ve výchozí pozici. Další tlačítko *Picture settings* umožňuje změnu nastavení kamery. Dvojice tlačítek *Play full test* a *Play last sequence* slouží, dalo by se říci, k náhledu právě definovaného testu. První z nich přehraje celý test tak, jak je v danou chvíli definován v levém panelu. Druhé tlačítko spustí pouze poslední skupinu bodů mezi kontrolními body. Poslední dvojice tlačítek *Save* a *Cancel* slouží pro uložení testu a druhé k zahození provedených změn.

Definování jednotlivých bodů se provádí jednoduše kliknutím do obrazu. Tím je aktivována metoda ClickPosition (3.5.8), která se postará o přidání bodu do panelu (obr. 3.11). Po kliknutí se také v místě kliknutí zobrazí černý čtvereček, který označuje místo kliknutí. Manipulátor se v definovaném místě dotkne a v té chvíli čtverec změní barvu. Po dokončení pohybu zmizí. Navíc se v okamžiku kliknutí uloží obraz kamery, do kterého je vykreslen bod doteku. Díky tomu je možné prohlédnout si definovaný test i bez připojení kamery a manipulátoru. Prohlížení obrazů testu je umožněno v okně náhledu testu (3.5.11), které se zobrazí po stisknutí tlačítka *Show* na kterémkoli z bodů v panelu. Stejně tak jako zobrazit je možné i kterýkoli bod smazat stisknutím tlačítka *Del*.




Obr. 3.10: Okno definice nového testu (3.5.7)


Obr. 3.11: Panel zobrazení bodu (3.5.9)


Obr. 3.12: Panel zobrazení kontrolního bodu (3.5.10)

### 3.5.8 Okno definice nového testu - popis metod

#### Metoda Form_NewTest_Load

*private void* Form_NewTest_Load(*object* sender, *EventArgs* e)

Metoda je spuštěna při otevření okna Form_NewTest (obr. 3.10). Zajišťuje připojení manipulátoru a obrazu kamery, nastaví objekt pro označování místa kliknutí v obrazu a událostem dokončení pohybu manipulátoru jsou přiřazeny obslužné metody. Pokud se okno spouští jako editace testu, pak seznam bodů není prázdný a metoda Add_Points_ToFLP (3.5.8) zajistí jejich zobrazení v postranním panelu. Vyprázdní se seznamy bodů nově přidaných a bodů pro smazání a ze souboru se načtou hodnoty rozsahů souřadnic, které je v vidět v záběru kamery a také body pro vykreslení obdélníku obrysu displeje.

#### Metoda ClickPosition

*private void* ClickPosition()

Metoda ClickPosition vstupní souřadnice několikrát transformuje. Prvním přepočtem jsou určeny souřadnice kliknutí uvnitř pictureBoxu. Následuje podmínka, která určí, zda se souřadnice kliknutí nacházení ve zobrazovaném obrazu. Pokud ano, pokračuje se přepočtem souřadnic z pictureBoxu na velikost obrazu kamery a pak přepočet na daný rozsah souřadnic, který kamera vidí. Dále se určí souřadnice středu obrazu a vzdálenost bodu od něj. Následuje korekce souřadnic podle zkreslení objektivu(1.3.3).

Dále se nastaví povolení kliknutí jako nepovoleno a zavolá se metoda, která označí bod v obrazu v aktuálním okně. Bod je přidán do seznamu bodů i do panelu metodou Add_Point (3.5.8) a robot odeslán na zadané souřadnice metodou

PerformClickLoader (3.2.1).

#### Metoda Add_Point

*public void* Add_Point(*double* X, *double* Y)

Nejprve přidá nový UserControl_Point (obr. 3.11), postupně se nastaví všechny jeho vlastnosti (properties) a nastaví se metody pro obsluhu událostí, které generuje. Kvůli jinému rozlišení kamery a obrazovky počítače je nutno přepočítat souřadnice bodu, aby mohl být správně vykreslen do obrazu. Po vykreslení je obraz uložen pod jménem nastaveným ve vlastnostech UserControl_Point. Nakonec je nově vytvořený bod přidán do flowLayotPanelu.

#### Metoda Add_Checkpoint

*public void* Add_Checkpoint()

Přidání kontrolního bodu probíhá obdobně jako přidávání normálního bodu (3.5.8), je ale jednodušší, jelikož se nemusí nic vykreslovat do obrazu. Pouze se vytvoří nový UserControl_Checkpoint (obr. 3.12), pořídí se obraz z kamery, uloží se pod stejným jménem jako vlastnost názvu obrazu, nastaví se metody pro obsluhu událostí a celý objekt se přidá do postranního panelu.

#### Metoda Del_Point

*private void* Del_Point(*object* sender, *EventArgs* e)

Pokud je ve chvíli aktivace povoleno mazání bodů, odstraní se daný UserControl z panelu a je přidán do seznamu bodů ke smazání. K samotnému smazání dojde až při uložení testu. Naopak, pokud je test uzavřen bez uložení, ke smazání nedojde a test tak zůstane beze změn.

#### Metoda Form_NewTest_Show_me

*private void* Form_NewTest_Show_me(*object* sender, *EventArgs* e)

Metoda je aktivována událostí generovanou UserControlem bodu nebo kontrolního bodu. Metoda otevře nové okno náhledu testu (obr. 3.13) a nastaví mu obraz podle odesílatele požadavku na otevření okna.

#### Metoda Add_Points_TofLP

*public void* Add_Points_TofLP()

Metoda projde všechny položky seznamu bodů a postupně přidá všechny UserControly do panelu. Přitom nastaví všechny jejich vlastnosti a metody obsluhující jejich události.

#### Metoda PlayToNextCheckpoint

*private void* PlayToNextCheckpoint()

Tato metoda se vyskytuje v programu vícekrát a zajišťuje, že bude na robot odeslána jedna sekvence bodů, od počátku seznamu po nejbližší kontrolní bod.

Postupně se prochází seznam bodů a ty se kopírují do seznamu pro odeslání. Když dojde na kontrolní bod, je procházení ukončeno, zpracované body jsou smazány a seznam bodů pro robot je odeslán metodou PerformClickLoader (3.2.1).

### 3.5.9 Panel bodu - popis metod

UserControl_Point (obr. 3.11) má dvě tlačítka a čtyři vlastnosti (property). Tlačítko *Del* je sice určeno ke smazání daného bodu a tlačítko *Show* pro zobrazení náhledu, ale ve skutečnosti obě tlačítka pouze generují události (eventy), které zpracovávají jim přiřazené metody v jiných třídách. Vlastnosti uchovávají souřadnice bodu X a Y a jméno souboru s obrazem pro daný bod. Vlastnosti jsou vyčteny při ukládání testu.

### 3.5.10 Panel kontrolního bodu - popis metod

U UserControl_Checkpoint (obr. 3.12) fungují obě tlačítka stejným způsobem jako u UserControl_Point (3.5.9). Rozdíl je pouze ve vlastnostech a to takový, že jedinou vlastností je jméno souboru obrazu pro náhled a porovnávání.

### 3.5.11 Okno náhledu testu

Okno náhledu testu (obr. 3.13) slouží pouze ke zobrazení a procházení obrazů pořízených během definice testu. Při spuštění okna se na pozadí předá celý seznam bodů, díky čemuž je možné prohlížet v okně celý test bez nutnosti okno neustále otvírat a zavírat. V okně jsou dvě tlačítka sloužící k posouvání náhledu vpřed a vzad. Pro posouvání je upraven i zobrazovací rámec. Kliknutím do něj způsobíme

také posun. V levé polovině vzad a v pravé polovině vpřed. V okně se také vypisují souřadnice bodů označených v obrazu.

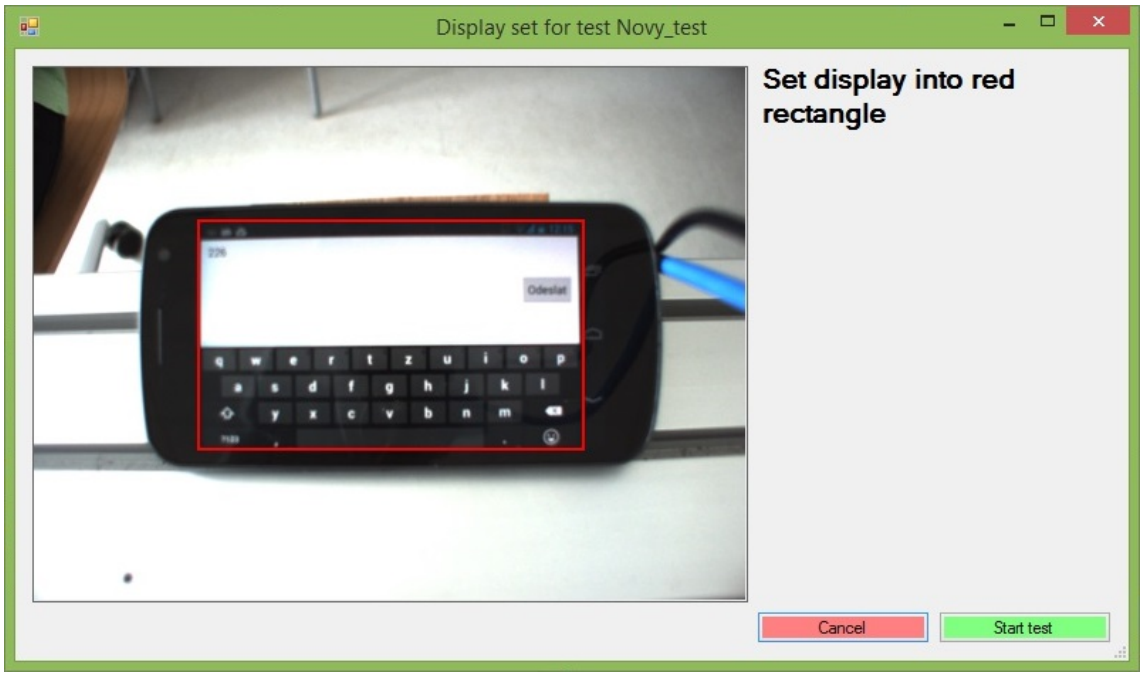

Obr. 3.13: Okno náhledu testu (3.5.11)

### 3.5.12 Okno přípravy spuštění testu

Okno přípravy spuštění testu (obr. 3.14) je zobrazován výhradně k jedinému účelu a to, aby byl po připojení ke kameře a manipulátoru nastaven testovaný displej do obdélníku vykresleného v obrazu. Zeleným potvrzovacím tlačítkem je spuštěno okno vyhodnocení testu (obr. 3.15).

### 3.5.13 Okno spuštění a vyhodnocení testu

Okno vyhodnocení testu (obr. 3.15) neobsahuje žádné další možnosti. Po zobrazení a připojení k robotu se automaticky spustí test. Postupně jsou manipulátorem projížděny všechny definované body, přičemž aktuálně prováděný úkon je vypsán v levém horní rohu jako Test progres. V každém z kontrolních bodů je pořízen obraz a je porovnáván s uloženým vzorem. Výsledek vyhodnocení je zobrazen jako UsecControl_Test (3.16), ve kterém jsou obrazy uloženého vzoru a aktuální snímek. Napravo od obrazů je vypsána míra shody obou obrazů.




Obr. 3.14: Okno přípravy spuštění testu (3.5.12)

Vyhodnocení zatím v programu není implementováno, ale je pro něj vše připraveno. Stačí pouze přidat třídu pro vyhodnocení a v kódu této třídy je již označen řádek pro volání metody porovnání obrazů.

### 3.5.14 Okno spuštění a vyhodnocení testu - popis metod

#### Metoda Form_RunningTest_Load

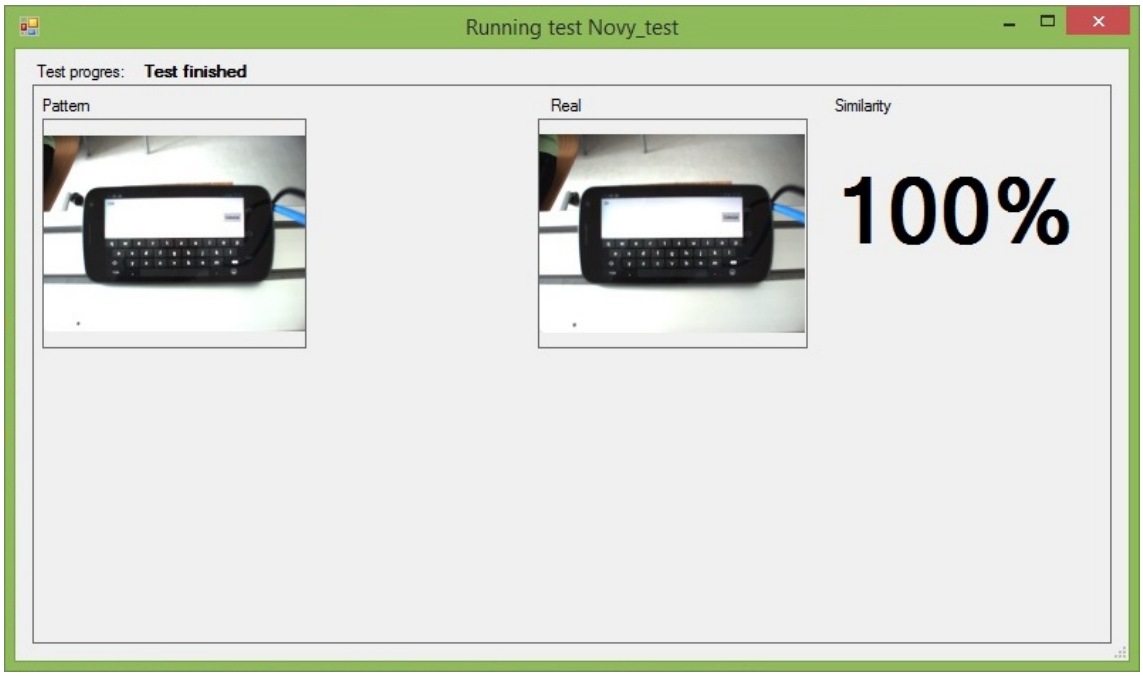
private void Form_RunningTest_Load(object sender, EventArgs e)

Při zobrazení dialogu je vyvolána tato metoda, která zajistí připojení kamery a spuštění přenosu obrazu, propojení robotu a nastavení metod obsluhujících události dokončení pohybu manipulátoru.

#### Metoda Form_RunningTest_VisibleChanged

void Form_RunningTest_VisibleChanged(object sender, EventArgs e)

Metoda je aktivována při změně viditelnosti okna. Pokud je okno viditelné, cyklem projdou postupně všechny body testu a je vytvořen seznam popisků určených pro Test progres.




Obr. 3.15: Okno vyhodnocení testu (3.5.13)


Obr. 3.16: Panel vyhodnocení testu (3.5.15)

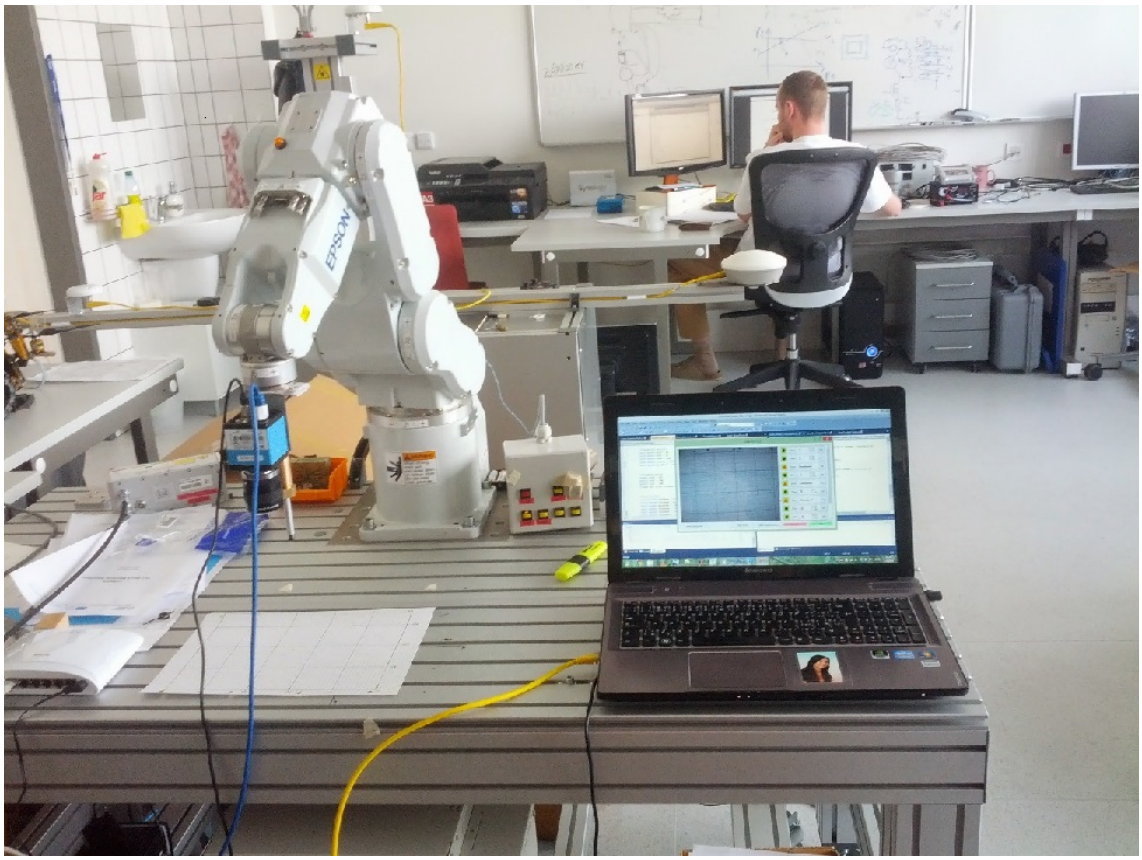
### Metoda RCinst_ClickDoneAll

private void RCinst_ClickDoneAll(object sender, EventArgs e)

Obsluha události dokončení všech pohybů manipulátoru. Pokud je metoda aktivována poprvé, znamená to, že robot dojel do výchozí polohy a je tedy spuštěn test aktivací metody PlayToNextCheckpoint, která funguje totožně jako stejnojmenná funkce třídy Form_NewTest (3.5.8). Při každé další aktivaci je jasné, že byl dosažen kontrolní bod a proto dojde k pořízení obrazu, porovnání se vzorem (prozatím bez porovnání) a přidání UserControl_Test (3.16) do okna vyhodnocení testu.

### 3.5.15 Panel vyhodnocení testu - popis metod

UserControl_Test (obr. 3.16) slouží pouze pro zobrazení výsledku testu a neobsahuje žádné aktivní prvky. Objekt obsahuje pouze tři vlastnosti, které lze jen nastavit. Dvě vlastnosti jsou určeny pro zobrazení porovnávaných obrazů, třetí pro nastavení popisku míry shody obou obrazů.




Obr. 3.17: Manipulátor s efektorem a kamerou při testování programu

# 4 ZÁVĚR

## 4.1 Zhodnocení dosažených cílů

V této práci je proveden rozbor robotických manipulátorů, možnosti jejich využití a jsou rozebrány principy funkce dotykových displejů. Je vytvořen funkční efektor, který reaguje na kapacitním i rezistivním displeji a zároveň je jeho konstrukce navržena tak, aby se co nejvíce snížilo riziko poškození ovládaného displeje. Je vytvořena také třída, která umožňuje ovládání manipulátoru a aplikace pro Windows a několik dalších tříd, které umožňují plánování trajektorie, její ukládání, editaci a spuštění. Před plánováním samotné trajektorie je možné definovat pracovní rovinu libovolně v pracovním prostoru manipulátoru.

## 4.2 Návrhy na pokračování projektu

Jako pokračování projektu by bylo jistě vhodné vylepšit efektor přidáním spínače do vnitřní konstrukce, který by plnil funkci stop tlačítka a pokud by byl stylus do těla efektoru zasunut příliš, robot by byl okamžitě vypnut a nemohlo by dojít k nechtěnému poškození testovaného zařízení.

V programu by bylo možné udělat také několik vylepšení ve vnitřní struktuře i v ovládáni, vylepšením stávajících prvků a přidáním některých nových. Velmi vhodná by byla úprava třídy RobotClicker a jejího využití tak, aby se po spuštění programu robot připojil a až do jeho ukončení by se neodpojoval. Nebylo by potom nutné neustále obnovovat komunikaci a připojovat robot za běhu programu. Ve vnitřní struktuře by bylo vhodné předělat metodu Load_Testlist() třídy Test_IO tak, aby v případě že neexistuje soubor, ze kterého má načítat data, návratovou hodnotou bude prázdný seznam namísto hodnoty null. Dalším potřebným vylepšením je, aby u všech bodů a kontrolních bodů, tedy objektů vycházejících z třídu UserControl, byla nastavena jejich společná vlastnost jména (Name), jako název typu bodu a bylo je od sebe možné rozlišovat prostým porovnáváním (podmínkou - if), namísto stávajícího vyhodnocení typu pomocí try a catch. Při definici nastavení testu by bylo vhodné, aby pracovní rovina byla přesně definována pomocí doteku třech bodů na displeji, vyčtením jejich souřadnic a výpočtem. Pro vyšší komfort zadávání by měl být pro definici výchozí pozice manipulátoru využít jiný objekt. Namísto rámečku s šipkami pro změnu hodnoty (numericUpDown) použít například posuvník (HScrollBar) nebo táhlo(TrackBar). V takovém případě by ale bylo nutné, aby stop tlačítko v efektoru nebo jiný systém chránil displej před zničením. Při opouštění okna definice a editace testu by mělo vyskočit okno s upozorněním, že chceme dané okno opustit bez uložení

změn a teprve po jeho potvrzení by mělo dojít k opuštění okna definice testu bez uložení provedených změn. Pro vyšší uživatelský komfort při definici testu by mohla být implementována funkce Drag'n'Drop pro přesouvání bodu v rámci jednoho testu prostým uchopením a tažením pomocí myši. Posledním z navrhovaných vylepšení je, aby u každého bodu testu bylo možné přidat prodlevu, tedy dobu, která musí uplynout před tím, než se robot dotkne v dalším bodě.

# LITERATURA


[1] PTFE - TECHNICKÉ INFORMACE. *Sittech.cz: Specialisté na těsnění* [online]. 2013, [cit. 2014. 5. 16]. Dostupné z URL: <http://www.sittech.cz/ptfe/teflon_informace.htm>.

[2] Neželezné kovy a jejich slitiny II. *Fsiforum.cz:06B-3SV.ppt* [online]. 2013, [cit. 2014. 5. 16]. Dostupné z URL: <http://www.fsiforum.cz/upload/soubory/databaze-predmetu/3SV/vechet/prednasky/06B-3SV.ppt>.

[3] Slovenské centrum produktivity. *Nadrozměrný portálový manipulátor* [online]. 2012, [cit. 2014. 5. 16]. Dostupné z URL: <http://www.slcp.sk/znalosti-online/images/inovacie/prax/nadrozmerny_portalovy_manipulator.jpg>.

[4] ŠOLC CSC., Doc. Ing. František a Ing. Luděk ŽALUD PH.D. Robotika. Skriptum. Vysoké učení technické v Brně. 2002, [cit. 2014. 5. 22].

[5] Epson C3 Compact 6-Axis Robot Manual. *Epson.com* [online]. 2009, [cit. 2014. 5. 16]. Dostupné z URL: <http://robots.epson.com/admin/uploads/product_catalog/files/EPSON_C3_Robot_Manual(R7).pdf>.

[6] FIREŠ, Martin. Demonstrační úloha pro robotický manipulátor Epson. Brno. Diplomová práce. Vysoké učení technické v Brně. Vedoucí práce doc. Ing. Luděk Žalud, Ph.D. *www.vutbr.cz* [online]. 2011, [cit. 2014. 5. 11]. Dostupné z URL: <https://www.vutbr.cz/studium/zaverecne-prace?action=detail&zp_id=52626&rok=&typ=&jazyk=&text=fires&hl_klic_slova=1&hl_abstrakt=0&hl_nazev=1&hl_autor=1&str=1>.

[7] FLORIÁN, Ing. Michal. Návrh paletizačního manipulátoru. Brno. Diplomová práce. Vysoké učení technické v Brně. Vedoucí práce Ing. Tomáš Kubela. *www.vutbr.cz* [online]. 2010, [cit. 2014. 5. 17]. Dostupné z URL: <https://www.vutbr.cz/studium/zaverecne-prace?action=detail&zp_id=30084&fid=&rok=&typ=&jazyk=&text=N%C3%A1vrh+paletiza%C4%8Dn%C3%ADho+manipul%C3%A1toru&hl_klic_slova=1&hl_abstrakt=0&hl_nazev=1&hl_autor=0&str=1>.

[8] Úvod do Matlabu. *https://cw.felk.cvut.cz* [online]. 2013, [cit. 2014. 5. 18]. Dostupné z URL: <https://cw.felk.cvut.cz/wiki/courses/a4m33mpv/cviceni/1_uvod/start>.

[9] Techbox. *Mobilnet.cz* [online]. 2013, [cit. 2014. 5. 18]. Dostupné z URL: <http://mobilenet.cz/clanky/techbox-dotykove-displeje---cim-se-lisi-rezistivni-od-kapacitniho-11566>.

[10] Jak fungují dotykové displeje. *Mobilmania.cz* [online]. 2014, [cit. 2014. 5. 18]. Dostupné z URL: <http://www.mobilmania.cz/uz-vim-jak-funguji-dotykove-displeje/a-1108570/default.aspx>.

[11] Imaging Source. *Theimagingsource.com* [online]. 2014, [cit. 2014. 5. 18]. Dostupné z URL: <http://http://www.theimagingsource.com/en_US/topics/gige-monochrome-cameras/>.

[12] DFK 51BG02H TIS. *scorpionvision.co.uk* [online]. 2014, [cit. 2014. 5. 18]. Dostupné z URL: <http://scorpionvision.co.uk/catalogue-index/industrial-cameras/the-imaging-source/tis-gige-ccd-mono-cameras/dfk-41bg02h-tis-gige-ccd-colour-camera-with-trigger-input-and-io-1>.

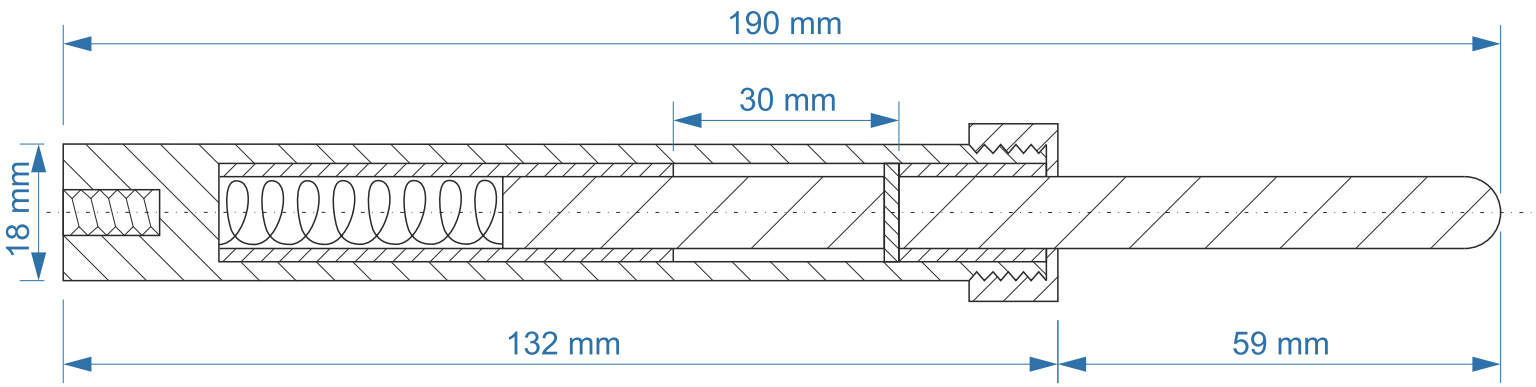
[13] Počítačové vidění. *midas.uamt.feec.vutbr.cz* [online]. 2010, [cit. 2014. 5. 18]. Dostupné z URL: <http://midas.uamt.feec.vutbr.cz/ZVS/zvs_cz.php>.

[14] CHROMÝ, Ing. Adam. 3D skenování pomocí proximitního planárního skeneru. Brno. Diplomová práce. Vysoké učení technické v Brně. Vedoucí práce doc. Ing. Luděk Žalud, Ph.D. *www.vutbr.cz* [online]. 2010, [cit. 2014. 5. 19]. Dostupné z URL: <https://www.vutbr.cz/studium/zaverecne-prace?action=detail&zp_id=66189&fid=&rok=&typ=2&jazyk=&text=chromy&hl_klic_slova=0&hl_abstrakt=0&hl_nazev=0&hl_autor=1&str=1>.

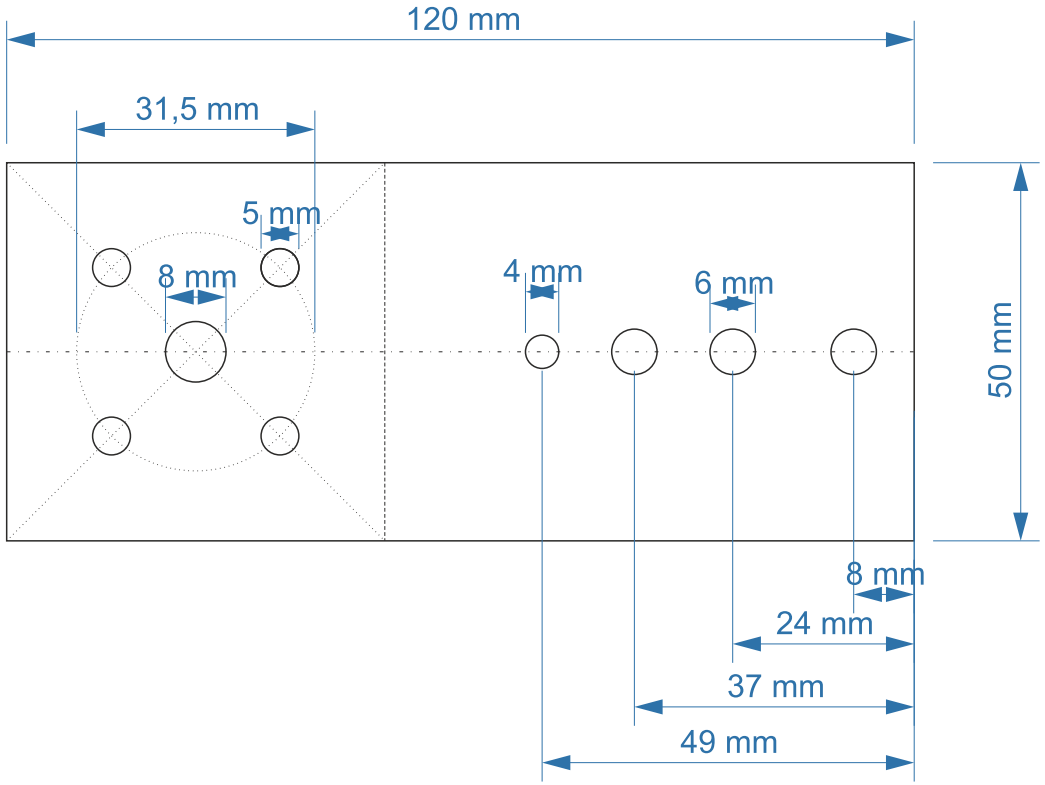
[15] DRHA, Štěpán. ŘÍZENÍ TŘÍOSÉHO KARTÉZSKÉHO MANIPULÁTORU. Brno. Bakalářská práce. Vysoké učení technické v Brně. Vedoucí práce Ing. Pavel Houška, Ph.D. *www.vutbr.cz* [online]. 2012, [cit. 2014. 5. 22]. Dostupné z URL: <https://www.vutbr.cz/studium/zaverecne-prace?action=detail&zp_id=50824&fid=&rok=&typ=&jazyk=&text=%C5%98%C3%8DZEN%C3%8D+T%C5%98%C3%8DOS%C3%89HO+KART%C3%89ZSK%C3%89HO+MANIPUL%C3%81TORU&hl_klic_slova=0&hl_abstrakt=0&hl_nazev=1&hl_autor=0&str=1>.

# SEZNAM PŘÍLOH

- A - Konstrukční výkres efektoru
- B - Konstrukční výkres držáku kamery
- C - CD-ROM




| | |
|---|---|
| | ONDREJ TATYREK |
| | EFEKTOR PRO EPSON C3 |
| | PRILOHA - A |




| | |
|---|---|
| | ONDREJ TATYREK |
| | DRZAK KAMERY |
| | PRILOHA - B |

# PŘÍLOHA C
# OBSAH CD-ROM:

- Elektronická podoba práce ve formátu PDF
- Výrobní výkresy pro efektor a držák kamery ve formátu PDF a EPS
- Zdrojové kódy ovládací aplikace
- Ovladač kamery Imaging Source